НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____ Олександр Коваль
«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Програмне забезпечення веб-технологій та мобільних пристроїв»

спеціальності 121 «Інженерія програмного забезпечення»

на тему: «розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри»

Виконав (-ла):
студент (-ка) IV курсу, групи ТІ-62
Коляда Олександр Андрійович _____

Керівник:
кандидат технічних наук, доцент,
Гагарін Олександр Олексагдрович _____

Консультант з Веб-програмування:
Посада, науковий ступінь, вчене звання,
Гагарін Олександр Олексагдрович _____

Рецензент:
Кандидат технічних наук, старший викладач,
Рачинський Артур Юрійович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки: 121 Інженерія програмного забезпечення

Спеціалізація: Програмне забезпечення веб-технологій та мобільних пристроїв

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____Олександр Коваль
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту

_____Коляді Олександру Андрійовичу_____
(прізвище, ім’я, по батькові)

1. Тема роботи розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри _____
_____
керівник роботи Гагарін Олександр Олексагдрович, кандидат технічних наук, доцент _____
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**

2. Строк подання студентом роботи ____
3. Вихідні дані до роботи_____

_____

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз інформаційних ресурсів
2. Дослідження технології розробки файлових сховищ
3. Вибір бази інструментів розробки
4. Інтерфейс користувача
5. Перелік ілюстративного матеріалу

_____

_____
_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ”___”_____201___ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	01.02.2020	
2.	Вивчення та аналіз задачі	20.02.2020	
3.	Розробка архітектури та загальної структури системи	20.04.2020	
4.	Розробка структур окремих підсистем	27.04.2020	
5.	Програмна реалізація системи	18.05.2020	
6.	Оформлення пояснювальної записки	01.06.2020	
7.	Захист програмного продукту	12.06.2020	
8.	Передзахист	12.06.2020	
9.	Захист	16.06.2020	

Студент _____ Коляда О.А._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Гагарін О.О._____
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

до бакалаврської дипломної роботи Коляди Олександра Андрійовича

на тему «розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри»

У дипломній роботі аналізується будова технологій для розробки інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри.

Метою даної дипломної роботи є вивчення проблеми будови і розробки компонентів для інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри: бази даних та інтерфейсу користувача для взаємодії з серверною частиною. Було вивчено особливості роботи з даними кафедри. На основі цих матеріалів реалізовані критерії вимог до програмного продукту, що розширює функції роботи з даними кафедри.

Під час реалізації роботи було розглянуто функціонал інструментів та методи вирішення проблем.

Результатом дипломної роботи є розроблені компоненти системи. Опис процесу розроблення компонентів: збудовано інтерфейс користувача задля управління системою для збереження необхідної інформації в необхідному сховищі даних.

Загальний обсяг роботи 79 сторінок.

Ключові слова: інтерфейс користувача, ресурси кафедри, навчальний план, база даних, CSS, HTML.

АННОТАЦИЯ

к бакалаврской дипломной работе Коляды Александра Андреевича

на тему: «разработка инструментальних Веб-способов организации доступа к информационным ресурсам кафедры»

В дипломной работе анализируется строение технологий для разработки инструментальных Веб-средств организации доступа к информационным ресурсам кафедры.

Целью данной работы является изучение проблемы строения и разработки компонентов для инструментальных Веб-средств организации доступа к информационным ресурсам кафедры: базы данных и интерфейса для взаимодействия с серверной частью. Были изучены особенности работы с данным кафедры. На основе этих материалов реализованы критерии требований к программному продукту, расширяет функции работы с данным кафедры.

При реализации работы были рассмотрены функционал инструментов и методы решения проблем.

Результатом работы является разработаны компоненты системы. Описание процесса разработки компонентов: построено интерфейс для управления системой для сохранения необходимой информации в необходимом хранилище данных.

Общий объем работы 78.

Ключевые слова: интерфейс, ресурсы кафедры, учебный план, база данных, CSS, HTML.

ANNOTATION

on Oleksandr Koliada bachelor's degree

thesis: " development of instrumental Web-methods of organizing access to information resources of the department "

The thesis analyzes the structure of technologies for developing instrumental Web-tools for organizing access to information resources of the department.

The aim of this work is to study the problems of the structure and development of components for instrumental Web-based tools for organizing access to the department's information resources: a database and an interface for interacting with the server part. The features of working with the department data were studied. Based on THESE materials, criteria for the requirements for a software product are implemented, it expands the functions of working with the department data.

When implementing the work, the functionality of the tools and methods for solving problems were considered.

The result of the work is the development of system components. Description of the component development process: an interface has been built for managing the system to store the necessary information in the necessary data warehouse.

The total amount of work is 78 pages.

Keywords: interface, department resources, curriculum, database, CSS, HTML.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І

ТЕРМІНІВ

БД – база даних.

ВНЗ – Вищий Навчальний Заклад.

СУБД – система управління базами даних.

НП – навчальний план.

ПП – програмний продукт.

XML – eXtensible Markup Language (розширювана мова розмітки).

JSON – JavaScript Object Notation (об’єктний запис JavaScript).

JSP – Java Server Page technology.

UX – User experience (користувацький досвід).

ВСТУП

Діяльність будь-якого закладу регулюється та залежить від певної кількості відповідних документів, які мають структурувати та контролювати усі робочі процеси, що відбуваються. Для вищого навчального закладу, зокрема, для кафедри факультету/інституту важливою складовою документації – є нормативні документи, які регулюють проведення навчальної роботи та її контроль, а також використовуються при складанні статистичної звітності, яка впливає на подальший перебіг процесів як кафедри, так і вищого навчального закладу.

Практика зберігання документів у сховищах широко поширилася за останні роки. Багато підприємств віддаляються від традиційного персонального комп'ютера с кількома вбудованими HDD, вважаючи віддалене сховище пріоритетним. Збільшення використання мобільних пристроїв змінює вимоги способу організації доступу до інформаційних ресурсів. Робітники бажають добувати доступ до файлів кафедри у будь-який час та з будь-яких девайсів.

Робітники тримають кафедральні дані на своїх власних девайсах, чи застосовують побічні сервіси. Ця практика може вивести до утрати даних кафедри при технічному збої. Також, інформаційні ресурси знаходяться у неструктурованому виді та розподілені по всяких девайсах. Це обтяжує доступ до файлів.

Дані мають бути надійно збереженими та мати змогу бути відновленими із резервної копії у будь-котрий час, задля запобігання витратам від неполадок в подачі електроенергії, зловмисних дій, стихійних лих та людської провини.

Тож, темою даної дипломної роботи є розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри

1 АНАЛІЗ ІНФОРМАЦІЙНИХ РЕСУРСІВ

1.1 Вступ

Розвиток вчення про інформацію розширило зміст суспільних ресурсів, додавши до них інформаційні ресурси. Але наявність інформаційних ресурсів поставило проблему їх репрезентації за допомогою процесів інформатизації, що базуються на інформаційних технологіях.

Ці технології мають високу наукоємність, пов'язані з частиною інтелектуального ресурсу, мають високу швидкістю впровадження інноваційних розробок в практику суспільного розвитку.

Вони не тільки ключовою бізнес-продукт, але і двигун науково-технічного прогресу, основа створення нових і вдосконалення існуючих технологічних процесів. Їх можна визначити як сукупність засобів, способів і методів вирішення практичних проблем, які спрямовані на задоволення потреб людей, стають пріоритетом розвитку людства, містять велику частку інтелектуального ресурсу, змінюють соціальну сферу і людини, ґрунтуються на інформаційних технологіях. Інформаційні технології базуються на інформаційних ресурсах конкретного суспільства, його сукупному громадському інтелекті.

Інформаційні ресурси - це наслідок інноваційного розвитку та використання інформації та знань. Тут простежується глибока причинно-наслідковий зв'язок інформації, інформаційних ресурсів, інформатизації та інформаційних технологій. Інформаційні технології - це сукупність предметних інформаційних знань і результатів інтелектуальної праці людини.

Ці результати обов'язково повинні бути зафіксовані на носіях будь-якого фізичного властивості, вони призначені для використання в інформаційному обороті. Але наявні в суспільстві інформаційні ресурси для їх позитивного інформаційного обороту вимагають своєї репрезентації. для цього необхідно

використовувати інформаційні технології, як інструмент процесів інформатизації. Інформатизація, в своїй суті і спрямованості, є організаційний науково-технічний і соціально-економічний процес забезпечення суспільства і особистості інформацією, яку мають інформаційні ресурси.

Інформатизація - це ядро структури формується інформаційної цивілізації, де інформація виступає особливим товаром, що користується підвищеним попитом у суспільства і фізичних осіб для задоволення їх потреб.

Найважливіше значення в структурі інформатизації належить її основного інструменту - інформаційних технологій, що представляють собою сукупність технічних, програмних і організаційно-економічних засобів, об'єднаних структурно і функціонально для вирішення конкретного завдання інформатизації.

Інформаційні технології - це комплекс взаємопов'язаних наукових, технологічних, інженерних наук, які вивчають методи ефективної організації праці людей, зайнятих обробкою і зберіганням інформації за допомогою обчислювальної техніки і методи організації і взаємодії з людьми і виробничим обладнанням, їх практичним застосуванням, а також пов'язані з усім цим соціальні, економічні і культурні проблеми.

Виходячи з вищевідзначеного аналізу змісту інформаційних технологій, можна стверджувати, що ці технології є система сукупних технічних засобів, способів, прийомів, методів обробки сукупних інформаційних ресурсів з метою створення унікальних продуктів і послуг, що володіють новою якісною визначеністю і не мають собі аналогів на світовому інформаційному ринку.

Оскільки інформаційні технології пронизують сферу інформаційних ресурсів (інформаційних продуктів та інформаційних послуг), отже вони вирішують завдання розвитку виробничої та невиробничої сфер. Це дозволяє ставити питання про їх класифікації.

У ній повинні бути відображені процеси створення і функціонування інформаційних технологій в різних сферах. У цю класифікацію необхідно віднести інформаційні технології:

- в сфері документації,
- науки,
- освіти,
- управління,
- менеджменту,
- програмного забезпечення,
- послуг,
- телекомунікацій,
- локальних і глобальних мереж.

Інформаційні технології розробляють операційні системи, охоплюють вищевідзначені сфери, особливо пов'язані з управлінням.

Операційні системи поділяються на одно програмні, що підтримують пакетні технології, тобто це пакетний режим обробки даних; багатопрограмні, що підтримують як пакетну технологію, так і діалогову технологію; розраховані на багато користувачів, що підтримують мережеву технологію. Мережева технологія забезпечує віддалену діалогову і пакетну технології.

1.2. Типи інформаційних ресурсів

В інформаційному суспільстві все більша увага приділяється не традиційним видам ресурсів (матеріальні, природні, трудові, фінансові, енергетичні), а інформаційним, які набувають першорядної важливості. Сьогодні оволодіння інформаційними ресурсами розглядається як економічна категорія.

Введення терміну "інформаційний ресурс" викликало досить багато дискусій. Інформаційні ресурси - це сукупність даних, організованих для отримання достовірної інформації в самих різних областях знань і практичної

діяльності. Законодавство Російської Федерації під інформаційними ресурсами має на увазі окремі документи і окремі масиви документів в інформаційних системах.

Ресурси визначають як запаси, джерела чого-небудь. У сучасному світі обсяг інформації збільшується лавиноподібна. Все важче стає вибрати з неї ту, яка найбільше відповідає існуючому запитом. В кінці XX століття виникає поняття "інформаційні ресурси".

Сьогодні поняття "інформаційні ресурси" досить багатопланово і включає в себе все різноманіття документів на традиційних і нетрадиційних носіях.

Інформаційні ресурси поділяються за класами, що збирається.

До первинно збирається, тобто тієї, яка відображає специфіку її джерела, області або сфери створення, виникнення, відноситься інформація, що утворюється самостійно в природних умовах (наприклад, кількість кіл на сиплі дерева, свідчить про його віці).

Інформація про кількісні та якісні характеристики різних соціальних процесів утворюють клас "знімається інформації". Виділені за цією ознакою інформаційні ресурси можна класифікувати як природні, виробничі, соціально-економічні. Наприклад, інформація про зростання населення.

Інший клас інформаційного ресурсу утворюють відомості, дані, одержувані штучно в процесі науково-дослідної діяльності, а також будь-якої творчої роботи. Вона базується на обробці вже наявної інформації за спеціальними параметрами і моделям (математична обробка, логічна, семантична). До цього ж класу відносяться і об'єкти, створені як авторські твори в галузі літератури, мистецтва.

Важливим компонентом цих ресурсів є інформація, що отримується в результаті інтелектуальної діяльності людини. Виділяється вторинна інформація, що виникає на основі переробки вже наявної інформації, і нова, яка фіксує те, що людство досі не знало. Сюди відносяться відкриття, прогнози в області різних соціальних і природних процесів.

До інформаційних ресурсів відносяться: бібліотеки, архіви, бази даних, ЗМІ тощо) і інформаційні сервіси (Інформаційні сервіси - це група сайтів, на яких можна скористатися різноманітними сервісними послугами: електронною поштою, блогом, а також познайомитися з механізмом його ведення, пошуком, різними каталогами, словниками, довідниками, прогнозом погоди, телепрограмою, курсами валют і т.д.

Розвиток світових інформаційних ресурсів дозволило:

Перетворити діяльність з надання інформаційних послуг (отримання і надання в розпорядження користувача інформаційних продуктів - сукупності даних, сформовану виробником для поширення в речовій або нематеріальній формі) в глобальну людську діяльність;

Сформувати світової і внутрішньодержавний ринок інформаційних послуг;

Утворити всілякі бази даних ресурсів регіонів і держав, до яких можливий порівняно недорогий доступ;

Підвищити обґрунтованість і оперативність прийнятих рішень у фірмах, банках, біржах, промисловості, торгівлі та ін. За рахунок своєчасного використання необхідної інформації.

За існуючою класифікацією, інформаційні ресурси можуть бути державними і недержавними і як елемент складу майна знаходяться у власності громадян, органів державної влади, органів місцевого самоврядування, організацій та громадських об'єднань.

1.3. Носії інформаційних ресурсів

Виходячи з визначення інформації як відомостей, з яких людина дістає знання для управління (цільова функція), ці відомості можуть мати як постійну форму (статичну), наприклад у вигляді підручника, викладеного на папері, або

змінну форму (динамічну), наприклад у вигляді звукового повідомлення, переданого людині людиною або радіо.

Таким чином, відомості як джерело інформації можуть бути закладені в природі, записані людиною або автоматом на будь-якому носії, перебувати на зберіганні або передаватися від джерела до одержувача (в цьому випадку прийнято говорити про повідомлення) і оброблятися з метою вилучення з них інформації.

З початку діяльності людини на землі його взаємовідносини один з одним і з навколишнім середовищем приносили величезний досвід і накопичуються знання, які виражаються у вигляді суб'єктивної і об'єктивної інформації. При цьому безпосередній досвід людини утворює первинну інформацію, або інформацію першого порядку.

Знання, отримані людиною в результаті обробки первинної інформації або в результаті деяких відомостей про наявний досвід інших людей, являє собою вторинну інформацію, або інформацію другого порядку. Таким чином, одержувані знання першого або другого порядків людиною фіксуються, зберігаються, обробляються і передаються іншим людям, що є основою виникнення інформаційної діяльності та інформаційних технологій.

Історія виникнення і розвитку інформаційних ресурсів показує динаміку розвитку людини як розумної істоти, здатного не тільки ефективно використовувати свої розумові і духовні здібності для спілкування між собою і активної взаємодії з природою, але також створювати методи і засоби фіксації, зберігання, обробки і передачі інформації і таким чином розвивати інформаційні технології і формувати інформаційне середовище для свого існування.

У процесі свого розвитку людство створювало і створює нові, все більш досконалі, механізми і технології виробництва, зберігання, обробки і передачі інформації для задоволення своїх інформаційних потреб і забезпечує тим самим формування безлічі інформаційних ресурсів.

1.4. Класифікація інформаційних ресурсів

Для класифікації інформаційних ресурсів і розбиття їх на певні види або категорії можна використовувати різноманітні ознаки. Найбільш узагальненою ознакою, що не вимагає аналізу ні семантичної, ні синтаксичної, ні прагматичної складових в інформаційні ресурси, служить ознака форми подання або фіксації інформації.

Всі інформаційні ресурси доцільно розбити на два класи: недокументовані, до якого відносять індивідуальні та колективні знання фахівців, і документовані.

Документовані інформаційні ресурси за ознакою закріплення інформації поділяють на текстові (письмові), графічні (креслення, схеми, графіки, карти, діаграми, картини), фото-, аудіо- (грамплатівки, аудіокасети і т. П.), Відео- (кінофільми, діапозитиви, слайди і т. п.) і електронні документи.

За ознакою фіксації інформації документовані інформаційні ресурси можна також розділити на два класи: зафіксована і зберігається на різного типу матеріальних носіях (різні матеріали: папір, полотно, глина, парафін, фотоплівка, кіноплівка, магнітна плівка і т. П.) І перетворена і зафіксована в електронному вигляді (пам'ять комп'ютера, дискета, компакт-диск і т.п.).

За ознакою автентичності документовані інформаційні ресурси підрозділяються на документи-оригінали і кооперування, або репродуковані, документи (мікрофільми, ксерокопії. Фотокопії і т.п.).

1.5. Роль і значення інформаційних ресурсів у розвитку інформаційних технологій і в інформатизації суспільства

Процес інформатизації суспільства, що сприяє задоволенню інформаційних потреб і реалізації прав фізичних, юридичних осіб, а також державних і громадських структур, включає:

- етап формування інформаційних ресурсів;

- етап подання інформаційних ресурсів і його передачі користувачу (споживачу);
- етап використання інформаційних ресурсів користувачем (споживачем);
- етап запиту користувачем (споживачем) на отримання необхідної йому інформації (нових знань).

Формування інформаційних ресурсів здійснюється в результаті інтелектуальної діяльності людини і представляється у вигляді документів, книг, статей, алгоритмів і програм, творів культури і мистецтва, зафіксованих на матеріальному носії, або у вигляді накопичених або накопичуються знань в мозку фахівця (вченого, лікаря, викладача, письменника, художника). На виході цього процесу виникає інформаційний продукт в речовій або нематеріальній формі, призначений для поширення, як і будь-який інший матеріальний продукт.

Створення інформаційного продукту відбувається за двома спонукальним причин. Перша - це генерація нових ідей і реалізація активного, ініціативного творчого начала людини - мислителя, вченого або художника. В даному випадку відсутній індивідуальний або суспільний запит або замовлення на отримання нових інформаційних ресурсів в конкретній предметній області знань.

Новостворений інформаційний продукт є основою розвитку, а в деяких випадках впливає кардинальним, стрибкоподібним чином на його розвиток (відкриття електрики, телефону, радіо, телебачення і т.д.).

2 ДОСЛІДЖЕННЯ ТЕХНОЛОГІЇ РОЗРОБКИ ФАЙЛОВИХ СХОВИЩ

2.1 Технології організації сховищ даних

Базою усіх сховищ є певний набір протоколів фізичного зберігання даних. Нині використовуються три головні класи фізичних моделей збереження даних.

2.1.1 DAS-система

Технологія Direct Attached Storage - особисте (пряме) підключення накопичувачів до сервера або до портативного комп’ютера. При цьому, накопичувачі можуть бути як внутрішніми, так і зовнішніми. Найпоширеніший випадок Direct Attached Storage-системи - це один диск всередині сервера або портативного комп’ютера. Також, до Direct Attached Storage-системі можна віднести будову внутрішнього Redundant Array of Independent Disks-масиву дисків з використанням Redundant Array of Independent Disks-Контролера.

Слід додати, що незважаючи на пряму можливість використовування терміна «Direct Attached Storage-системи» у відношенні до одиночного диску або до внутрішніх масивів дисків, під Direct Attached Storage-системою треба розуміти зовнішню стійку з дисками, яку дозволено розглядати як автономну систему захисту даних. Окрім самостійного живлення, автономні Direct Attached Storage-системи мають спеціальний контролер (мікропроцесор) для правління масивом накопичувачів. В якості подібного контролера може виступати Redundant Array of Independent Disks-контролер з можливістю полем Redundant Array of Independent Disks-масивів різноманітних рівнів.

Також слід зазначити те, що автономні Direct Attached Storage-системи можуть мати декілька зовнішніх потоків введення та виведення, що дозволяє можливість прямого підключення до Direct Attached Storage-системі декількох портативних комп'ютерів водночас.

Інтерфейсами до підключення внутрішніх накопичувачів або зовнішніх накопичувачів в технології Direct Attached Storage можуть виступати такі інтерфейси як Small Computer Systems Interface, Serial AT Attachment, Parallel AT Attachment та Fibre Channel. Якщо Small Computer Systems Interface інтерфейси, Serial AT Attachment і Parallel AT Attachment можуть застосовуватися здебільш для підключення тільки внутрішніх накопичувачів, то такий інтерфейс як Fibre Channel може служити виключно для підключення тільки зовнішніх накопичувачів та автономних систем захисту даних. Пріоритет Інтерфейсу Fibre Channel тут полягає як раз в тому, що Fibre Channel не має грубого обмеження по довжині та може також використовуватися в такому випадку, коли сервер або портативний комп'ютер, що підключається до Direct Attached Storage-системі, знаходиться від Direct Attached Storage-системи на значній відстані. Такі Інтерфейси як Small Computer Systems Interface та Serial AT Attachment теж можуть застосовуватися для включення зовнішніх систем захисту даних (в такому випадку інтерфейс Serial AT Attachment звуть eSATA), втім вони мають дуже суворе обмеження по найбільшій довжині кабелю, що з'єднує Direct Attached Storage-систему і включається до серверу.

До головних переваг Direct Attached Storage-систем можна також віднести їх дуже малу вартість (в зрівнянні з деякими іншими вирішеннями систем захисту даних), безпосередність розгортання і адміністрування, а також дуже високу стрімкість обміну даними поміж системою зберігання та сервером. Власне кажучи, саме з оцієї основи вони стали дуже популярними в сегменті малих штатів і середніх корпоративних мереж. В той же час Direct Attached Storage-системи мають і свої власні недоліки. У першу чергу це дуже висока ціна зберігання і адміністрування даними внаслідок їх розгалуження по організації, і також примусовий простій самої мережі у моменти додавання подальших дисків та першочергова необхідність накопичування пам'яті або мікропроцесорної та процесорної сили сервера при перевищенні декотрого розміру простору диску. Переобтяженість між-мережевого трафіку з додаванням інших серверів дуже обтяжує проблеми захисту даних, а також

перешкоджає оперативному використанню багатьох ресурсів і т.д. Витрати та інші проблеми зростають дуже швидко.

У теперешній час Direct Attached Storage-системи займають першочергове положення, однак частина таких систем безвідривно скорочується, та на заміну їм поступають або універсальні вирішення з можливостяси легкої міграції до Network Attached Storage-систем, або системи, які передбачають перспективу їх використання у якості Direct Attached Storage-, Network Attached Storage - і навіть Storage Area Network-систем.

2.1.2 NAS-система

Network Attached Storage-системи - це мережеві порядки збереження даних, які підключаються до мережі також, як і мережевий портативний комп'ютер у місцевій або світовій мережі, котрий дає користувачам обчислювальні та дискові ресурси, роутер чи якесь інакше мережеве приладдя. Network Attached Storage-системи видають собою еволюцію файлових портативних комп'ютерів у місцевій або світовій мережі, котрі дають користувачам обчислювальні та дискові ресурси. Щоб розібрати розбіжність поміж звичайним файловим сервером та Network Attached Storage-пристроєм, підсумуємо те, що традиційний файловий портативний комп'ютер у місцевій або світовій мережі, котрий дає користувачам обчислювальні та дискові ресурси. Задля збереження ресурсів можна використовувати жорсткі диски, які поміщаються в портативний комп'ютер у місцевій або світовій мережі, котрий дає користувачам обчислювальні та дискові ресурси, чи підключенням до серверу Direct Attached Storage-пристрою. Налагодження файлового сервера робиться за поміччю серверного операційного порядку. Цей підхід до реалізації системи збереження даних сьогоденням є найбільшим та доступним у сегменті середніх локальних мереж, проте є один серйозний мінус. Універсальний портативний комп'ютер у місцевій або світовій мережі, котрий дає користувачам обчислювальні та дискові ресурси ніяк не малоцінне вирішення. Також

більшість багатофункціональних перспектив, характерних універсального портативного комп'ютера у місцевій або світовій мережі, котрий дає користувачам обчислювальні та дискові ресурси, в файловому сервері не використовується. Тож треба зробити оптимізований файловий сервер із оптимальною операційною системою та збалансованою формою. Це і роблять у собі Network Attached Storage-пристрої, котрі у цьому сенсі можуть роздивлятись як прості файлові сервери.

Окрім оптимізованої операційної системи, позбавленої від робот, не зв'язаних із обслугою файлової системи та виконанням введення та виведення даних, Network Attached Storage-системи мають багатофункціональну по бистроті шляху файлову систему. Network Attached Storage-системи розробляються так, щоб вся ця обчислювальна інтенсивність фокусується тільки на функціях обслуги та збереження файлів. Операційна система знаходиться в флеш пам'яті і установлюється продуцентом. Підключення Network Attached Storage-пристроїв до мережі та налаштування є достатньо простою задачею та під силу багато якому навченому користувачу.

У зрівнянні із сакраментальними файловими портативними комп'ютерами у місцевій або світовій мережі, котрі дають користувачам обчислювальні та дискові ресурси, Network Attached Storage-пристрої бувають здебільш багатофункціональними та дешевшими. Сьогодення майже усі Network Attached Storage-пристрої націлені на використовування в мережах Ethernet (Gigabit Ethernet, Fast Ethernet) на базисі протоколів Transmission Control Protocol / Internet Protocol. Шлях до пристроїв Network Attached Storage здійснюється завдяки спеціальним протоколам переходу до файлів. Найбільше розповсюдженими протоколами багато файлового доступу є протоколи Common Internet File System та Network File System.

2.1.3 SAN-система

Storage Area Network - це спеціальна мережева технологія задля збереження даних. Всі ці мережі робляться у виді часткових спеціальних мереж до складу місцевої (local area network) або світової (wide area network) мережі.

SAN-мережі зв'язують один чи декілька серверів (SAN-серверів) з поодинокими чи кількома апаратами збереження даних. Storage Area Network-мережі позволяють кожному Storage Area Network-серверу здобувати шлях до кожного приладу збереження даних, не загружаючи сторонні сервери та місцеву мережу. Також, дозволений розмін даними поміж приладами збереження даних без втручання серверів. Storage Area Network-мережі позволяють великій кількості користувачів зберегти інформацію у одному місці та воєдино користуватися нею. Redundant Array of Independent Disks-масиви, різноманітні бібліотеки, і ще Just a bunch of disks-системи (масиви дисків, які не з'єднані у Redundant Array of Independent Disks) можуть використовуватися як прилади збереження даних.

2.1.4 Апаратна віртуалізація

Технологія віртуалізації замінила огляд нинішніх сховищ даних. Подібно тому, як фізичні носії перетворилися у віртуальні машини, фізичні сховища перетворюються у віртуальні диски. Застосовуючи технології віртуалізації монітор віртуальних машин постачає емуляцію апаратного оточення задля будь-якої віртуальної машини, залучаючи пам'ять, комп'ютер та зберігання. Базові нинішні рішення застосовують емуляцію місцевих фізичних дисків ніби спосіб додати сховище задля будь-якої віртуальної машини. Подібно тому, як базовою часткою збереження в Direct Attached Storage є фізична машина, базовою часткою в пам'яті віртуального диску є віртуальна машина. Таклж, віртуальні диски обглядаються не як самостійні об'єкти, а і як частка визначеної віртуальної машини, таким же чином, як і місцеві диски є часткою персонального комп'ютера.

Розробка віртуальних дисків в образі макета збереження блочного сховища Direct Attached Storage -стилі, на вершині Network Attached Storage або Storage Area Network, показує одну із дуже вагомих характеристик нинішнього збереження центру обробки даних.

Оскільки потік даних із віртуальної машини переходить з програмного забезпечення до комп'ютерного програмного забезпечення чи обладунку процесора, що забезпечує одночасне і паралельне виконання декількох віртуальних машин, на кожній з яких виконується власна операційна система, а не до апаратних приладів у шинах пристроїв, протокол, що застосовується у віртуальній машині задля обміну даними із комп'ютерним програмним забезпеченням чи обладунком процесору, що забезпечує одночасне і паралельне виконання декількох віртуальних машин, на кожній з яких виконується власна операційна система, не має відповідати протоколу застосованому задля зв'язку із приладом збереження.

Оце приводить до розділу поміж моделлю збереження, що рухається угору до віртуальної машини, та протоколом, що застосовується комп'ютерним програмним забезпеченням чи обладунком процесору, що забезпечує одночасне і паралельне виконання декількох віртуальних машин, на кожній з яких виконується власна операційна система, задля збереження даних, які дають гнучкість перемішувати та з’єднувати протоколи зберігання та моделі збереження даних, та навіть швидко міняти протокол збереження з забезпекою для віртуальних машин.

Обрання розробки повноцінно прозоре для застосунку, тому що у заключному підсумку усі протоколи виглядатимуть нарівно для адміністратора та віртуальної машини. Вони виглядатимуть так, як місцеві фізичні диски. Тож, розроблювач програмного застосунку у більшій частині інфраструктури публічних хмар не зможе бачити, котрий протокол збереження застосовується. Протокол може мінятися з динамічною швидкістю.

Від розподілу поміж макетом збереження та протоколом збереження, протокол для збереження виповняється інфраструктурною задачею, у першу чергу дуже вагомим задля ціни та продуктивності. Функціональні резерви подають програмно-визначні вирішення.

2.1.5 Сховища програмно-визначного типу

Сховище програмно-визначного типу (Software-defined storage) - це подальший хід в піднесенні програмного продукту по будові збереження даних задля керування обслуговуваннями збереження на базисі бізнес-орієнтованих принципів самостійно від програмного забезпечення.

Задум застосовування систем збереження даних, у котрих програмне забезпечення від’єднано від програмних засобів виявляється інтересним задля середніх підприємств адже подає їм перспективу змагатися із більшими підприємствами.

Зате Сховища програмно-визначного типу різняться від віртуальних сховищ, котрі позволяють купі приладів збереження об’єднуватися у окремий віртуальний прилад.

Сховища програмно-визначного типу відбирають від програмного забезпечення весь функціонал замість ємкості. Сховища програмно-визначного типу можуть представлятися також як і програмна система, що дає обслуговування збереження даних.

Сховища програмно-визначного типу позволяють об'єднувати підручні апаратні ресурси фірми у монолітну інфраструктуру, що дає шанс міняти свої функції у відповідності попиту фірми.

Відгороджуючи апаратне забезпечення сховищ від програмного забезпечення що управляє ними, сховища програмно-визначного типу позволяють фірмам застосовувати загальнодоступне неоднорідне апаратне забезпечення.

2.2 Категорії сховищ даних

Бувають три базові типи сховищ: об’єктне, блочне та файлове.

2.2.1 Сховище даних файлового типу

Файлове сховище великих даних являє собою створений на порталі елемент, який посилається на дані розташування, доступні на вашому ArcGIS GeoAnalytics Server. Розташування файлового сховища великих даних може використовуватися в якості вхідних і вихідних даних для векторних даних (точок, полілініій, полігонів і табличних даних) інструментів геоаналітікі. Файлові сховища великих даних дозволяють розбивати набори даних на розділи, зберігаючи здатність роботи з декількома такими розділами як з єдиним набором даних.

Використання файлового сховища великих даних для вихідних даних дозволяє зберігати результати в форматах, які можна використовувати для інших робочих процесів, таких як файл parquet для подальшого аналізу або зберігання.

При створенні файлового сховища великих даних генерується елемент на вашому порталі. Елемент вказує на сервіс каталогу великих даних, який описує набори даних у файловому сховищі великих даних і їх схему, включаючи геометрію і інформацію про час, а також вихідні формати, звані шаблонами, які ви зареєстрували. При використанні файлового сховища великих даних в якості вхідних даних в інструменті ArcGIS GeoAnalytics Server, ви можете вказати шлях до цього елементу і запустити аналіз для набору даних.

Існує кілька причин, чому краще використовувати файлове сховище великих даних. Ви можете зберігати свої дані в доступному місці до тих пір, поки не будете готові виконати аналіз. Файлове сховище великих даних дозволяє працювати з даними під час виконання аналізу, тому ви можете

продовжувати додавати дані в набір, що знаходиться в файловому сховищі великих даних, без необхідності перереєстрації або опублікування своїх даних.

Ви також можете змінити маніфест, щоб видалити, додати або оновити набори даних у файловому сховищі великих даних. Файлове сховище великих даних відрізняється надзвичайною гнучкістю з точки зору визначення геометрії та часу і допускає кілька форматів часу в окремому наборі даних.

Файлові сховища великих даних дозволяють розбивати набори даних на розділи, зберігаючи здатність роботи з декількома такими розділами як з єдиним набором даних. Використання файлового сховища великих даних для вихідних даних дозволяє зберігати результати в форматах, які можна використовувати для інших робочих процесів, таких як файл parquet для подальшого аналізу або зберігання.

Щоб підготувати дані для файлового сховища великих даних необхідно представити набори даних вкладеними папками окремої батьківської папки, яка буде зареєстрована. У цій реєструється батьківської папці імена вкладених папок будуть збігатися з іменами наборів даних.

Якщо ці вкладені папки міститимуть кілька підпапок або файлів, то весь вміст цих вкладених папок вищого рівня буде вважатися окремим набором даних, і до них буде застосовуватися та ж схема. При реєстрації батьківської папки всі підкаталоги зазначеної папки також реєструються на сервері GeoAnalytics Server. Завжди треба реєструвати батьківську папку (наприклад, \\ machinename \ FileShareFolder), що містить один або кілька підпапок окремих наборів даних.

2.2.2 Сховище даних блочного типу

Блокові системи називаються так тому, що збережені в них дані розбиваються на блоки однакового розміру. Блок даних - це не файл, що не закінчений об'єкт, а просто якісь дані в шматку фіксованого розміру. Файл даних може бути розміщений в кінцевому числі блоків, і якщо останній з цих

блоків залишається незаповненим, то він все одно матиме той же фіксований розмір, що і заповнені блоки. Сервер отримує доступ до цих блокам через мережу зберігання даних (storage attached network).

Блокова система зберігання даних може бути використана будь-якій операційній системою як дисковий том цієї системи. Кожному сервера може бути надано практично необмежений розмір диска. Користувач бачить цей диск в інтерфейсі операційної системи сервера. Операційна система сервера може підключатися до блокам даних через високошвидкісні інтерфейси Fiber Channel або iSCSI. Тому основна перевага блокових систем - високу швидкодію.

Операційна система сервера може бути як фізичної (хост), так і віртуальної (віртуальна машина). Слід зазначити, що фізичних серверів потрібні додаткові контролери для того, щоб отримувати доступ до вихідних блокам.

Розмір блоків визначається виробником обладнання. Однак архітектор, який розробляє дизайн корпоративної IT-системи, може визначати розмір блоку даних в залежності від вимог конфігурації і сценарію застосування. Розмір блоку може залежати і від типу фізичних дисків в системах зберігання (жорсткі диски HDD або флеш-накопичувачі SSD), а також від типу самих даних клієнтів. Правильний вибір розміру диска в багатьох випадках допомагає оптимізувати параметри роботи корпоративної IT-системи в цілому.

Операційна система сервера присвоює кожному блоку даних простий ідентифікатор розташування (location ID), за яким його можна швидко знайти в storage attached network, саме тому блокова система зберігання даних має високу швидкодію.

Зауважимо, що, наприклад, в об'єктних системах зберігання в якості такого ідентифікатора використовуються метадані (тип файлу, назва файлу, дата та ін). Зрозуміло, що при такому механізмі час пошуку об'єкта в системі збільшується, хоча об'єктні системи зберігання мають багато переваг для інших ситуацій.

Зазвичай блокові системи зберігання вибирають, коли при проектуванні IT-системи критичними є такі параметри, як швидкість введення-виведення даних IOPS (input output operations per second), а також затримка (latency) доступу до даних через SAN. Основна ідея об'єктного сховища полягає в тому, щоб супроводити файл довільного змісту якимись додатковими параметрами: метаданими, що описують об'єкт.

Поєднання будь-якого файлу з його метаданими можна розглядати як комп'ютерний об'єкт і застосовувати до його обробці відповідні, вже усталені, методи. Наприклад, це можуть бути сервери баз даних або бізнес-критичних додатків, що вимагають високої швидкодії і високої надійності.

У таких базах даних, як Oracle або SAP Hana, використовуються саме блокові системи. Віртуальні машини VMware також за замовчуванням використовують блочне зберігання, оскільки при цьому можна побудувати швидкодіючу і багатофункціональну віртуальну інфраструктуру. Блочне зберігання може бути також використано практично у всій архітектурі, будь то одиничний сервер або ж високонадійні кластерні серверні рішення.

Блокові системи зберігання зараз є безумовним лідером за часткою в корпоративному зберіганні даних як в малих підприємствах, так і в IT-системах великих корпорацій.

Блочне зберігання даних - дуже популярна послуга хмарних провайдерів. Вона дає можливість використовувати таку корисну якість, як еластичність. Тобто користувачу немає необхідності закуповувати обладнання для СГД з «запасом на майбутнє». Він може замовити у провайдера ємність зберігання відповідно до поточних потреб і потім швидко заказати її при необхідності. У більшості випадків цей процес може зайняти кілька хвилин.

При цьому дані будуть добре захищені високою доступністю послуги хмарного зберігання і технологією реплікації в хмарі. Крім того, шифрування, яке забезпечує хмарний провайдер, не дозволить іншим серверам, що використовують ту ж хмарну СГД, отримати доступ до чужих даних. Таким чином, коли потрібно швидко і недорого розширити наявне блочне сховище,

використання послуги блокової СГД в хмарі може бути оптимальним варіантом.

Якщо говорити не про публічну, а про приватну хмару ІТ-системи або дата-центру великої організації, то і тут можна використовувати подібні механізми блочного зберігання. При цьому замість різних замовників, дані яких потрібно розділяти в хмарі, виступатимуть підрозділи (відділи, сервіси, філії) цієї організації. І тут використання блочного зберігання найчастіше буде кращим варіантом, оскільки ця технологія забезпечує високу швидкість бізнес-процесів організації, а також достатню гнучкість перерозподілу ресурсів в масштабах організації.

2.2.3 Сховище даних об’єктного типу

Дискові і файлові сховища використовуються вже десятиліттями. Їх технологія достатньо відпрацьована і добре відома. У них можна розмістити будь-які прикладні файли. Число цих файлів може бути дуже великим, але все-таки воно обмежене. Основна ідея об'єктного сховища полягає в тому, щоб супроводити файл довільного змісту якимись додатковими параметрами: метаданими, що описують об'єкт. Поєднання будь-якого файлу з його метаданими можна розглядати як комп'ютерний об'єкт і застосовувати до його обробці відповідні, вже усталені, методи.

Не завжди є можливість використання одного файлу різними користувачами і різними додатками одночасно. Не завжди є можливість зручного управління доступом до цих файлів. Як правило, інтерпретація даних в них повністю залежить від використовують їх програм, так як ніякого опису структури даних ні в самому файлі, ні в файловому сховищі немає. Зате операції з дисковими і файловими сховищами найшвидші. Правда, за умови, що ці сховища знаходяться в локальній або добре пов'язаної мережі.

Системи управління базами даних дозволяють структурувати дані шляхом їх розміщення в пов'язаних між собою таблицях. Наявність індексів і

механізму транзакцій робить обробку інформації швидкої і надійної. Однак ми все частіше і частіше стикаємося з необхідністю обробляти неструктуровані або слабо структуровані дані, наприклад, фотографії, аудіо- та відеозаписи та, взагалі, будь-які файли з довільним змістом.

Для розміщення неструктурованих даних добре підходять об'єктні сховища, які забезпечують досить універсальну обробку таких даних.

Основна ідея об'єктного сховища полягає в тому, щоб супроводити файл довільного змісту якимись додатковими параметрами: метаданими, що описують об'єкт. Поєднання будь-якого файлу з його метаданими можна розглядати як комп'ютерний об'єкт і застосовувати до його обробці відповідні, вже усталені, методи.

Кожен об'єкт може бути описаний багатьма різнорідними метаданими. Наприклад, в них можуть міститися докладні характеристики якогось відео фрагменту або фотографії.

Ідея розглядати довільний файл як об'єкт виникла не вчора. Вже досить давно багато СУБД підтримують такий тип даних як BLOB (Binary Large OBject). Однак зберігання файлів, особливо великих, в реляційній базі даних навряд чи слід визнати раціональним підходом.

Метадані файлів в об'єктному сховище можуть бути проіндексовані. Це прискорить пошук потрібного об'єкта за ознаками, що містяться в метаданих.

Як було сказано, в кожному об'єкті міститься тільки один файл. Однак на утримання цього файлу не накладається ніяких обмежень. Тобто їм може бути архівний файл з безліччю вкладених файлів або будь-яка інша структури, що включає в себе різні прикладні фрагменти.

Розмір одного об'єкта обмежений. Але якщо буде потрібно зберігати файл більшого обсягу, його можна розділити на сегменти.

Сучасні об'єктні сховища можуть містити в собі величезну кількість об'єктів.

Другою важливою особливістю об'єктних сховищ є те, що в переважній більшості випадків вони призначені для взаємодії з ними через інтернет.

Більшість об'єктних сховищ є розподіленими. Частини сховища, рознесені апаратно або навіть географічно, називаються зонами.

Надійне зберігання об'єктів серед іншого забезпечується тим, що кожен його екземпляр перебудовується в інші зони. Підтримується груповий доступ до контейнерів та об'єктів з урахуванням прав користувачів. І як правило, взаємодіють з ними не люди, а додатки або інформаційні системи. Основою прикладного інтерфейсу (API) є протокол HTTP.

На відміну, наприклад, від систем управління базами даних, об'єктні сховища не призначені для обробки даних всередині себе. Тому основними операціями з ними є розміщення і отримання об'єктів.

Так як в сховищі може одночасно перебувати велика кількість об'єктів, керувати ними без можливості угруповання було б важко. Один із способів розподілу об'єктів - так звані, контейнери, які можна пов'язувати з окремим проектом або інформаційної підсистемою клієнта.

Деякі системи можуть підтримувати версію об'єктів. Захоплюватися нею без особливої потреби не варто, так як вона збільшує споживання дискового простору.

Для забезпечення спільної роботи різних користувачів передбачена система управління доступом до контейнерів і об'єктам. При необхідності вміст об'єкта може бути зашифровано.

Об'єктні сховища не передбачають ієрархії об'єктів і груп об'єктів, подібної до тієї, яку мають файлові системи у вигляді системи вкладених папок. Але назва об'єкта може містити в собі різні символи, в тому числі «/», що дозволяє імітувати ієрархічний шлях до об'єкта.

Більшість об'єктних сховищ є розподіленими. Частини сховища, рознесені апаратно або навіть географічно, називаються зонами. Надійне зберігання об'єктів серед іншого забезпечується тим, що кожен його екземпляр перебудовується в інші зони. Підтримується груповий доступ до контейнерів та об'єктів з урахуванням прав користувачів.

2.3 Метадані

Концепція метаданих проста і в той же час складна. Ми без праці розуміємо, що таке дані: це інформація, якою ми обмінюємося, яку обробляємо і споживаємо в постійно розвивається цифровому суспільстві. Дані, особливо цифрові, можуть приймати різні форми. Звичайні розмови, текстові повідомлення або соціальні мережі - все це способи передачі даних. Цифровий банк або торгові транзакції мають на увазі передачу даних. Веб-вміст, цифрові і потокові розваги, бази даних або інформаційні сховища будь-якого роду - це приклади публікації даних.

Метадані описують суть цих даних: вони надають інформацію про ці дані. Все дуже просто. Але якщо копнути трохи глибше, ми виявимо, що «опис» даних являє собою складну з технічної точки зору завдання і одночасно соціально-політичну проблему.

Метадані - це засіб класифікації, упорядкування та характеристики даних або вмісту. Національна організація з інформаційних стандартам (NISO) пропонує класифікацію, яку можна застосувати для всіх типів даних або сховищ даних, від бібліотек до веб-сайтів, для текстових та нетекстових даних, в цифровий або матеріальній формі.

NISO описує три типи метаданих:

- Описові метадані включають таку інформацію, як точки контакту, заголовок або автор публікації, анотація роботи, які використовуються в роботі ключові слова, географічне розташування або навіть пояснення методології. Ці дані служать для виявлення, збору або групування ресурсів за загальними для них характеристикам. Щоб зрозуміти, як описові метадані співвідносяться з інформаційними даними, відвідайте сторінки Ділові та споживчі дослідження Європейської комісії з економіки та фінансів. Крім даних досліджень ви можете отримати Метадані BCS по дослідженню кожної з країн

Євросоюзу, наприклад Франції. Файли метаданих містять контактні дані, опис методології і дату кожного дослідження, але в них немає питань і відповідей, отриманих в ході дослідження.

- Структурні метадані пояснюють склад або організацію ресурсів. Наприклад, цифрову книгу можна публікувати у вигляді зображень окремих сторінок, файлу PDF або HTML. Ці сторінки або компоненти зазвичай групують в глави. Дані про голів, зміст або відомості про макеті сторінок вважаються структурними метаданими. До структурних метаданих відносяться також такі записи, як структурна карта сторінок або інших ресурсів веб-сайту, подія вторгнення або записи відомостей про голосові дзвінки.
- Адміністративні метадані використовуються для управління ресурсом. Дати створення або отримання, права доступу, права або походження, або правила утилізації, такі як зберігання або видалення, є прикладами прав, які може застосовувати цифровий архівіст, куратор. Подібні метадані виявляться корисними для адміністратора бази даних або для адміністраторів, що відповідають за отримання даних з трафіку телекомунікаційних мереж або мереж передачі даних, або журналів систем безпеки або даних про події.

Познайомившись з різними типами метаданих, ви можете оцінити, наскільки корисні вони можуть бути для будь-яких компаній, організацій або державних органів, які займаються збором, управлінням або зберіганням метаданих у великих масштабах. Ви можете також зрозуміти, що дії зі збору метаданих в великих масштабах можуть стати джерелом полеміки. Концепція метаданих проста і в той же час складна. Ми без праці розуміємо, що таке дані: це інформація, якою ми обмінюємося, яку обробляємо і споживаємо в постійно розвивається цифровому суспільстві.

Дані, особливо цифрові, можуть приймати різні форми. Звичайні розмови, текстові повідомлення або соціальні мережі - все це способи передачі даних. Цифровий банк або торгові транзакції мають на увазі передачу даних. Веб-

вміст, цифрові і потокові розваги, бази даних або інформаційні сховища будь-якого роду - це приклади публікації даних.

2.4 Висновки

У цьому пункті була досліджена техніка, що застосовується задля будови нинішніх систем збереження та обміну даними. Було досліджено базисні різновиди фізичних моделей, що застосовуються задля будови апаратного сховища даних. Також було вивчено використовування наборів віртуалізації задля розподілу програмних приладів керування розподіленим сховищем та апаратними техніками збереження даних.

У ознаці базису задля будови програмного продукту обрані програмно обумовлені сховища, тому що вони роблять велику частку потрібного функціонування із правління поділом сховища та дуже добре годяться задля сформульованих завдань.

Також було помічено перспективи застосовування метаданих задля оптимальної роботи зі сховищем даних. Файлові метадані можна зберігати в виді пошукових індексів, які надають перспективу опрацьовувати дуже тяжкі запити пошуку.

3 ВИБІР БАЗИ ІНСТРУМЕНТІВ РОЗРОБКИ

3.1 Вступ

Для того, щоб забезпечити якісну веб-розробку, можна користуватися різними інструментами. Сьогодні деякі з майстрів до цих пір продовжують використовувати самий звичайний блокнот, однак це вже не дуже зручно, так як робота через це стає вельми повільної. В сучасних умовах розробляти сайти необхідно якомога швидше, робити це якісно і в максимально стислі терміни.

Для цього стали з'являтися різні інструменти, розраховані для веб-розробки. З їх допомогою можна забезпечити більш ефективну і продуктивну роботу.

Для створення веб-сторінки можна використовувати різні інструменти від простих до найскладніших. Вибір інструментів залежить від того, кому і для якої мети створюється сайт (сайти фірм створюють професійні фахівці за допомогою дорогих програм, сайт зроблений руками любителя створюється за допомогою простих інструментів). Простий програмою може бути, наприклад, текстовий редактор Notepad, до числа складних належать MS Expression Web і Adobe DreamWeaver.

Прості і дешеві інструменти не завжди просто використовувати. Для того щоб створити веб-сторінку за допомогою Notepad, необхідно дуже добре знати мову HTML, що при використанні Expression Web зовсім не обов'язково. Для створення веб-сторінки можна використовувати різні інструменти від простих до найскладніших. Вибір інструментів залежить від того, кому і для якої мети створюється сайт (сайти фірм створюють професійні фахівці за допомогою дорогих програм, сайт зроблений руками любителя створюється за допомогою простих інструментів). Простий програмою може бути, наприклад, текстовий редактор Notepad, до числа складних належать MS Expression Web і Adobe DreamWeaver. Тут необхідність використання мови HTML залежить від налаштованого комплекту інструментів.

Зрозуміло, сучасні веб-розробники не могли б працювати по-справжньому ефективно, якби не використовували потужні програмні інструменти, що полегшують роботу, і роблять її більш швидкою і досконалою. Отже, наведемо список деяких програм, які використовуються в процесі розробки і створення сайтів.

Dreamweaver - вельми зручна і популярна програма, яка дозволяє швидко створювати сторінки, використовуючи як режим редагування HTML, так і режим візуального редагування контенту (а також змішаний режим, який можна застосовувати, якщо є монітор відповідного розміру).

Photoshop - потужна програма, яка давно зарекомендувала себе як найкращий платний редактор растрової графіки. Дозволяє редагувати картинки, а також зберігати їх у форматі, відповідному для web.

Notepad++ - Компактний редактор HTML з підсвічуванням коду. Завдяки підсвічуванню і деяким додатковим функціям робота по редагуванню коду зазнає суттєвого спрощення.

3.2 HTML

HTML (HyperText Markup Language) - це мова розмітки документів. Він застосовується в усьому світі. Браузер інтерпретує код HTML для відображення його на комп'ютері, планшеті або телефоні. В першу чергу мова HTML призначався для обміну науковими документами. Верстка документів відбувається за допомогою спеціальних дескрипторів (але найчастіше їх називають тегами). На даний момент остання версія - HTML5. Для спрощення і зручності було введено поняття «гіпертекст». Гіперпосилання (або просто посилання), є частиною гіпертексту, і вона посилається на інший HTML документ.

HTML-документи - це файли, які закінчуються розширенням .html або .htm. Ви можете переглядати його за допомогою будь-якого веб-браузера (наприклад, Google Chrome, Safari або Mozilla Firefox). Браузер читає HTML-

файл і відображає його вміст, щоб користувачі інтернету могли його переглядати.

Зазвичай середній веб-сайт включає кілька різних HTML-сторінок (англ). Наприклад: домашні сторінки, звичайні сторінки, сторінки контактів матимуть окремі HTML-документи.

Кожна HTML-сторінка складається з набору тегів (також званих елементами), які ви можете назвати будівельними блоками веб-сторінок. Вони створюють ієрархію, яка структурує контент за розділами, параграфами, заголовкам і іншим блокам контенту.

Більшість елементів HTML мають відкриття і закриття, в яких використовується синтаксис <tag> </ tag>.

Для форматування тексту використовуються багато вбудовані теги. HTML-теги мають два основних типи: блок-рівень і вбудовані теги.

HTML не є мовою програмування, тобто він не має можливості створювати динамічні функції. Замість цього він дозволяє організовувати і форматувати документи, аналогічно Microsoft Word.

HTML є основною мовою розмітки в інтернеті. Він запускається спочатку в кожному браузері і підтримується консорціумом World Wide Web.

Ви можете використовувати його для створення структури контенту веб-сайтів і веб-додатків. HTML-документи - це файли, які закінчуються розширенням .html або .htm. Ви можете переглядати його за допомогою будь-якого веб-браузера (наприклад, Google Chrome, Safari або Mozilla Firefox). Браузер читає HTML-файл і відображає його вміст, щоб користувачі інтернету могли його переглядати. Це найнижчий рівень технологій frontend, який служить основою для стилізації, яку ви можете додати за допомогою CSS і функціональності, яку ви можете реалізувати за допомогою JavaScript.

Як і більшість речей, HTML має як сильні сторони так і слабкі.

Плюси:

- Широко використовується мова з великою кількістю ресурсів і величезним співтовариством.

- Виконується спочатку в кожному веб-браузері.
- Поставляється з плоскої кривої навчання.
- У відкритому доступі і абсолютно безкоштовний.
- Чистий і послідовна розмітка.
- Офіційні веб-стандарти підтримуються консорціумом World Wide Web (W3C).
- Легко інтегрується з базовими мовами, такими як PHP і Node.js.

Мінуси:

- В основному використовується для статичних веб-сторінок. Для динамічної функціональності вам може знадобитися використовувати JavaScript або бекенда-мову, такий як PHP.
- Це не дозволяє користувачеві реалізувати логіку. В результаті все веб-сторінки потрібно створювати окремо, навіть якщо вони використовують одні і ті ж елементи, наприклад. заголовки і колонтитули.
- Деякі браузери приймають нові функції повільно.
- Іноді поведінка браузера важко передбачити (наприклад, старі браузери не завжди створюють нові теги).

HTML був побудований так, що сторінки відображалися на всіх пристроях однаково. Пізніше додали графічне оформлення (CSS).

3.3 CSS

CSS - це формальна мова, службовець для опису оформлення зовнішнього вигляду документа, створеного з використанням мови розмітки (HTML, XHTML, XML). Назва походить від англійського Cascading Style Sheets, що означає «каскадні таблиці стилів».

Призначення CSS - відокремлювати те, що задає зовнішній вигляд сторінки, від її змісту. Якщо документ створено тільки з використанням HTML, то в ньому визначається не тільки кожен елемент, але і спосіб його

відображення (колір, шрифт, положення блоку). Необхідність розробки CSS було визнано консорціумом W3C в 1990-х роках. У 1996 році був прийнятий стандарт CSS1, що дозволяє змінювати параметри шрифту, колір, атрибути тексту, вирівнювання і відступи. У 1998 році відбувся вихід CSS2, який додав можливості використання блокової верстки, звукових таблиць, що генерується змісту, покажчиків, сторінкових носіїв. Якщо ж підключені каскадні таблиці стилів, то HTML описує тільки черговість об'єктів. А за всі їх властивості відповідає CSS. В HTML досить прописувати клас, не перераховуючи всі стилі кожен раз.

Така технологія:

- забезпечує відносно просту і швидку розробку, тому що одного разу створене оформлення можна застосовувати до багатьох сторінок;
- підвищує гнучкість і зручність редагування - досить винести правку в CSS, щоб оформлення змінилося всюди;
- робить код більш простим, знижуючи повторюваність елементів. Його простіше читати програмістам і пошуковим роботам;
- прискорює час завантаження, тому що CSS може керуватися при першому відкритті, а в наступних зчитуються тільки структура і дані;
- збільшує кількість візуальних рішень для подання вмісту;
- забезпечує можливість легко застосовувати до одного документу різні стилі (наприклад, створювати адаптовану версію для мобільних пристроїв або спеціальні стилі для людей з вадами зору).

Необхідність розробки CSS було визнано консорціумом W3C в 1990-х роках. У 1996 році був прийнятий стандарт CSS1, що дозволяє змінювати параметри шрифту, колір, атрибути тексту, вирівнювання і відступи. У 1998 році відбувся вихід CSS2, який додав можливості використання блокової верстки, звукових таблиць, що генерується змісту, покажчиків, сторінкових носіїв.

Версія CSS3 помітно збільшила можливості стилів: стало доступним створення анімований елементів без використання JavaScript, з'явилася

підтримка згладжування, тіней, градієнтів і т. Д. Специфікація була розділена на модулі, кожен з яких став розвиватися окремо. З 2011 року ведеться розробка модулів CSS4. Можливості поки описані в чорнових варіантах.

Селектор повідомляє, до якого елементу будуть застосовані описувані в CSS властивості стилю. Як селектора може виступати будь-який тег, якому задається форматування (розмір, колір і т. Д.). Якщо для тега потрібно задати різні стилі або застосувати один для відмінних елементів, використовуються класи і запис виду «Тег.Клас {властивість: значення;}». Ім'я класу задається латиницею, може містити підкреслення або дефіс. Блок оголошень складається з пар «властивість: значення» (запис завжди черга двокрапка), розміщених в фігурних дужках.

Записи закінчуються крапкою з комою. CSS нечутливий до табуляції, прогалин, регістру. Вибір способу запису (стовпчиком з відступом або просто в рядок) залишається на розсуд розробника. Якщо для одного селектора прописані різні значення для одного властивості, то пріоритет віддається нижньої записи. Якщо не вказувати тег, а починати запис з «.Клас», то можна використовувати правило для будь-якого тега.

Якщо перераховувати кілька класів для одного тега, до нього застосовується все описані стилі. Ідентифікатор задає унікальне ім'я елемента для зміни стилю або звернення за допомогою скрипта. Запис «# Ідентифікатор {властивість: значення;}». Назва ідентифікатора складається з букв латинського алфавіту, допустимо використовувати дефіс і підкреслення. Щоб застосувати ідентифікатор до конкретного тегу, вказується його ім'я, потім без пробілу і через знак решітки назва ідентифікатора.

Блок оголошень складається з пар «властивість: значення» (запис завжди черга двокрапка), розміщених в фігурних дужках. Записи закінчуються крапкою з комою. CSS нечутливий до табуляції, прогалин, регістру. Вибір способу запису (стовпчиком з відступом або просто в рядок) залишається на розсуд розробника. Якщо для одного селектора прописані різні значення для одного властивості, то пріоритет віддається нижньої записи.

Зазначені типи стилів можуть спокійно існувати один з одним, якщо вони не намагаються змінити вид одного елемента. У разі виникнення суперечності спочатку має пріоритет стиль користувача, потім стиль автора і останнім йде стиль браузера.

3.4 JavaScript

Javascript - це мова програмування. Це означає те, що з його допомогою можна користуватися всіма основними можливостями мов програмування.

Спочатку JavaScript був створений, щоб «зробити веб-сторінки живими». Програми на цій мові називаються скриптами. Вони можуть вбудовуватися в HTML і виконуватися автоматично при завантаженні веб-сторінки. Скрипти поширюються і виконуються, як простий текст. Їм не потрібна спеціальна підготовка або компіляція для запуску. Це відрізняє JavaScript від іншої мови - Java.

Коли JavaScript створювався, у нього було інше ім'я - «LiveScript». Однак, мова Java був дуже популярний в той час, і було вирішено, що позиціювання JavaScript як «молодшого брата» Java буде корисно. Згодом JavaScript став повністю незалежним мовою зі своєю власною специфікацією, яка називається ECMAScript, і зараз не має ніякого відношення до Java.

Сьогодні JavaScript може виконуватися не тільки в браузері, а й на сервері або на будь-якому іншому пристрої, який має спеціальну програму, що називається «движком» JavaScript.

У браузера є власний движок, який іноді називають «віртуальна машина JavaScript».

Різні движки мають різні «кодові імена». наприклад:

- V8 - в Chrome і Opera.
- SpiderMonkey - в Firefox.
- «Trident» і «Chakra» для різних версій IE

- «ChakraCore» для Microsoft Edge, «Nitro» і «SquirrelFish» для Safari і т.д.

Сучасний JavaScript - це «безпечна» мова програмування. Він не надає низько-рівневі доступи до пам'яті або процесору, тому що спочатку був створений для браузерів, які не потребують цього. JavaScript - це єдина технологія браузера, що поєднує в собі три речі: повна інтеграція з HTML / CSS. Прості речі робляться просто. Підтримується всіма основними браузерами і включений за замовчуванням. Ось що робить JavaScript особливим. Ось чому це найпоширеніший інструмент для створення інтерфейсів в браузері. Хоча, звичайно, JavaScript дозволяє робити програми не тільки в браузерах, але і на сервері, на мобільних пристроях і т.п.

Можливості JavaScript сильно залежать від оточення, в якому він працює. Наприклад, Node.JS підтримує функції читання / запису довільних файлів, виконання мережевих запитів і т.д.

У браузері для JavaScript є все, що пов'язано з маніпулюванням веб-сторінками, взаємодією з користувачем і веб-сервером.

Наприклад, в браузері JavaScript може:

- Додавати новий HTML-код на сторінку, змінювати існуючий вміст, модифікувати стилі.
- Реагувати на дії користувача, клацання миші, перемістити вказівник, натискання клавіш.
- Відправляти мережеві запити на віддалені сервера, завантажувати і завантажувати файли (технології AJAX і COMET).
- Задавати питання відвідувачеві, показувати повідомлення.
- Запам'ятовувати дані на стороні клієнта («local storage»).

JavaScript - це єдина технологія браузера, що поєднує в собі три речі: повна інтеграція з HTML / CSS. Прості речі робляться просто. Підтримується всіма основними браузерами і включений за замовчуванням. Ось що робить JavaScript особливим. Ось чому це найпоширеніший інструмент для створення

інтерфейсів в браузері. Хоча, звичайно, JavaScript дозволяє робити програми не тільки в браузерах, але і на сервері, на мобільних пристроях і т.п.

Можливості JavaScript в браузері обмежені заради безпеки користувача. Мета полягає в запобіганні доступу недобросовісної веб-сторінки до особистої інформації або нанесення шкоди даними користувача.

Приклади таких обмежень включають в себе:

- JavaScript на веб-сторінці не може читати / записувати довільні файли на жорсткому диску, копіювати їх або запускати програми. Він не має прямого доступу до системних функцій ОС.
- Сучасні браузери дозволяють йому працювати з файлами, але з обмеженим доступом, і надають його, тільки якщо користувач виконує певні дії, такі як «перетягування» файлу в вікно браузера або його вибір за допомогою тега <input>.
- Існують способи взаємодії з камерою / мікрофоном і іншими пристроями, але вони вимагають явного дозволу користувача.

- Різні вікна / вкладки не знають один про одного. Іноді одне вікно, використовуючи JavaScript, відкриває інше вікно. Але навіть в цьому випадку JavaScript з однієї сторінки не має доступу до іншої, якщо вони прийшли з різних сайтів (з іншого домену, протоколу або порту).
- JavaScript може легко взаємодіяти з сервером, з якого прийшла поточна сторінка. Але його здатність отримувати дані з інших сайтів / доменів обмежена. Хоча це можливо в принципі, для чого потрібно явне згоду (виражене в заголовках HTTP) з віддаленої стороною. Знову ж таки, це обмеження безпеки.

JavaScript спочатку створювався тільки для браузера, але зараз використовується на багатьох інших платформах. Сьогодні JavaScript займає унікальну позицію в якості самого поширеного мови для браузера, який володіє повною інтеграцією з HTML / CSS. Багато мов можуть бути «передані» в

JavaScript для надання додаткових функцій. Рекомендується хоча б коротко розглянути їх після освоєння JavaScript.

3.5 PHP

PHP (рекурсивний акронім словосполучення PHP: Hypertext Preprocessor) - це поширена мова програмування загального призначення з відкритим вихідним кодом. PHP спеціально сконструйований для веб-розробок та його код може впроваджуватися безпосередньо в HTML.

Замість рутинного виведення HTML-коду командами мови (як це відбувається, наприклад, в Perl або C), скрипт PHP містить HTML з вбудованим кодом. Код PHP відділяється спеціальними початковим і кінцевим тегами <? Php i?>, Які дозволяють "перемикатися" в "PHP-режим" і виходити з нього.

PHP відрізняється від JavaScript тим, що PHP-скрипти виконуються на сервері і генерують HTML, який надсилається клієнту. Якби на сервері був розміщений скрипт, подібний вищенаведеного, клієнт отримав би тільки результат його виконання, але не зміг би з'ясувати, який саме код його справив. Ви навіть можете налаштувати свій сервер таким чином, щоб звичайні HTML-файли оброблялися процесором PHP, так що клієнти навіть не зможуть дізнатися, чи отримують вони звичайний HTML-файл або результат виконання скрипта.

Значною відмінністю PHP від будь-якого коду є те, що виконується на стороні клієнта, наприклад, JavaScript, є те, що PHP-скрипти виконуються на стороні сервера. PHP відрізняється від JavaScript тим, що PHP-скрипти виконуються на сервері і генерують HTML, який надсилається клієнту.

Якби на сервері був розміщений скрипт, подібний вищенаведеного, клієнт отримав би тільки результат його виконання, але не зміг би з'ясувати, який саме код його справив. Ви навіть можете налаштувати свій сервер таким чином, щоб звичайні HTML-файли оброблялися процесором PHP, так що клієнти навіть не зможуть дізнатися, чи отримують вони звичайний HTML-

файл або результат виконання скрипта. Ви навіть можете налаштувати свій сервер таким чином, щоб HTML-файли оброблялися процесором PHP, так що клієнти навіть не зможуть дізнатися, чи отримують вони звичайний HTML-файл або результат виконання скрипта.

PHP дозволяє створювати якісні Web-додатки за дуже короткі терміни, отримуючи продукти, легко модифікуються і підтримувані в майбутньому. PHP простий для освоєння, і в той же час здатний задовольнити запити професійних програмістів.

Можливості PHP дуже великі. Головним чином, область застосування PHP сфокусована на написання скриптів, що працюють на стороні сервера; таким чином, PHP здатний виконувати все те, що виконує будь-яка інша програма CGI. Наприклад, обробляти даних форм, генерувати динамічні сторінки, відсилати і приймати cookies. Але PHP здатний виконувати і безліч інших завдань.

Існують три основні області, де використовується PHP:

- Створення скриптів для виконання на стороні сервера. PHP найбільш широко використовується саме таким чином. Все, що вам знадобиться, це парсер PHP (у вигляді програми CGI або серверного модуля), вебсервер і браузер. Щоб Ви могли переглядати результати виконання PHP-скриптів в браузері, вам потрібен працюючий вебсервер і встановлений PHP.
- Створення скриптів для виконання в командному рядку. Ви можете створити PHP-скрипт, здатний запускатися незалежно від веб-сервера та браузера. Все, що вам потрібно - парсер PHP. Такий спосіб використання PHP ідеально підходить для скриптів, які повинні виконуватися регулярно, наприклад, за допомогою cron (на платформах nix або Linux) або за допомогою планувальника завдань (Task Scheduler) на платформах Windows. Ці скрипти також можуть бути використані в задачах простої обробки текстів.

- Створення додатків GUI, що виконуються на стороні клієнта. Можливо, PHP є не найкращим мовою для створення подібних додатків, але, якщо ви дуже добре знаєте PHP і хотіли б використовувати деякі його можливості в своїх клієнт-додатках, ви можете використовувати PHP-GTK для створення таких додатків. Подібним чином Ви можете створювати і крос-платформні додатки. PHP-GTK є розширенням PHP і не поставляється разом з дистрибутивом PHP.

PHP доступний для більшості операційних систем, включаючи Linux, багато модифікації Unix (такі, як HP-UX, Solaris і OpenBSD), Microsoft Windows, Mac OS X, RISC OS, і багатьох інших. (Існує навіть версія PHP для OS / 2. Невідомо, правда, наскільки відповідна нинішнім реаліям).

Також в PHP включена підтримка більшості сучасних веб-сервера, таких, як Apache, Microsoft Internet Information Server, Personal Web Server, серверів Netscape і iPlanet, сервера Oreilly Website Pro, Caudium, Xitami, OmniHTTPd і багатьох інших. Для більшості серверів PHP поставляється в якості модуля, для інших, що підтримують стандарт CGI, PHP може функціонувати в якості процесора CGI.

3.6 Технологія File Share

Обмін файлами (File sharing) - це практика розповсюдження або надання доступу до цифрових медіа, таких як комп'ютерні програми, мультимедіа (аудіо, зображення та відео), документи або електронні книги. Обмін файлами може бути досягнуто різними способами. Загальні методи зберігання, передачі та розповсюдження включають обмін вручну, використовуючи знімні носії інформації, централізовані сервери в комп’ютерних мережах, гіперпосилання документів на базі всесвітньої веб-сторінки та використання розподілених одноранговых мереж.

Обмін файлами peer-to-peer базується на архітектурі додатків peer-to-peer (P2P). Спільні файли на комп'ютерах інших користувачів індексуються на серверах каталогів. Технологію P2P використовували популярні сервіси, такі як Napster та LimeWire.

Служби синхронізації та обміну файлами на основі хмарних технологій реалізують автоматизовану передачу файлів, оновлюючи файли із спеціалізованого каталогу спільного використання на всіх мережевих пристроях кожного користувача. Файли, розміщені в цій папці, також, як правило, доступні через веб-сайт та мобільний додаток, і їх можна легко ділити з іншими користувачами для перегляду чи співпраці.

Такі сервіси стали популярними через послуги, орієнтовані на споживача файлового хостингу, такі як Dropbox та Google Drive. З наростаючою потребою легко обмінюватися великими файлами в Інтернеті, з'явилися нові платформи відкритого доступу, що додають ще більше послуг до їх основного бізнесу (хмарне зберігання, синхронізація на декількох пристроях, онлайн-співпраця), такі як ShareFile, Tresorit, WeTransfer, MeeroDrop або Hightail.

Обмін файлами передбачає використання технології, яка дозволяє користувачам Інтернету обмінюватися файлами, розміщеними на їх окремих комп'ютерах. Програми однорангових (P2P) програм, наприклад, які використовуються для обміну музичними файлами, є одними з найпоширеніших форм технології обміну файлами. Однак програми P2P вводять ризики для безпеки, які можуть поставити під загрозу вашу інформацію чи комп'ютер.

Встановлення зловмисного коду - програми P2P часто використовуються зловмисниками для передачі шкідливого коду. Під час використання цих програм важко, якщо не неможливо, перевірити, чи джерело файлів є надійним; Зловмисники можуть включати у файли шпигунські програми, віруси, троянські коні чи черв'яків. Під час завантаження файлів комп'ютер заражається.

Визначення конфіденційної та особистої інформації - використовуючи програми P2P, ви можете надавати іншим користувачам доступ до особистої інформації. Незалежно від того, чи доступні певні каталоги чи тому, що ви надаєте персональну інформацію тому, що вважаєте довіреною особою чи організацією, несанкціоновані люди можуть мати доступ до ваших фінансових чи медичних даних, особистих документів, конфіденційної корпоративної інформації чи іншої особистої інформації. Після того, як інформація потрапляє до сторонніх людей, важко дізнатися, скільки людей зверталися до неї. Наявність цієї інформації може збільшити ризик крадіжки особистих даних.

Сприйнятливість до атаки. Деякі програми P2P можуть попросити вас відкрити певні порти брандмауера для передачі файлів. Однак, відкриття деяких із цих портів може надати зловмисникам доступ до вашого комп'ютера або дозволити їм атакувати ваш комп’ютер, скориставшись будь-якими вразливими місцями, які можуть існувати в додатку P2P. Є деякі програми P2P, які можуть змінювати і проникати між брандмауерами, без вашого відома.

Відмова в обслуговуванні - завантаження файлів викликає значну кількість трафіку по мережі. Ця діяльність може зменшити доступність певних програм на вашому комп'ютері або обмежити доступ до Інтернету.

Звинувачення - Файли, якими надаються програми P2P, можуть включати піратське програмне забезпечення, захищені авторським правом матеріали або порнографію. Якщо ви завантажуєте їх, навіть невідомо, вам можуть загрожувати штрафи або інші судові дії. Якщо ваш комп'ютер перебуває в мережі компанії і викриває інформацію про клієнтів, ви і ваша компанія можуть нести відповідальність.

Найкращий спосіб усунути ці ризики - уникати використання додатків P2P. Однак, якщо ви вирішили використовувати цю технологію, ви можете дотримуватися деяких хороших практик безпеки, щоб мінімізувати ризик:

- використання та підтримка антивірусного програмного забезпечення - антивірусне програмне забезпечення розпізнає та захищає ваш комп'ютер від більшості відомих вірусів. Однак зловмисники постійно

пишуть нові віруси, тому важливо тримати своє антивірусне програмне забезпечення актуальним.

- встановити або ввімкнути брандмауер - брандмауери можуть уникнути деяких видів зараження, блокуючи шкідливий трафік, перш ніж він зможе потрапити на ваш комп’ютер. Деякі операційні системи насправді містять брандмауер, але вам потрібно переконатися, що він увімкнене.

4 ІНТЕРФЕЙС КОРИСТУВАЧА

4.1 Вступ

Інтерфейс забезпечує комфортну взаємодію між користувачем і технікою. При розробці інтерфейсу програми слід враховувати ергономіку і пристрої, з якими буде взаємодіяти додаток (портативний комп'ютер, планшет або смартфон). Користувач (його комфорт і зручність) повинен бути головним критерієм у побудові інтерфейсу і створенні дизайну додатків.

Під інтерфейсом користувача розуміється сукупність інформаційної моделі проблемної області, засобів і способів взаємодії користувача з цією моделлю, а також компонентів, що забезпечують формування ІМ в процесі роботи програмної системи. Інтерфейс грає найважливішу роль в "приживлюваності" розроблених програмних продуктів. Завдання проектування інтерфейсу користувача моноваріантне, при цьому процедура вибору в значній мірі неформальна.

Алгоритми додатку в сучасному підході відрізняються від алгоритмів програми більш складними обчисленнями з можливим формуванням проміжних об'єктів. У традиційному підході формується алгоритм прямого перетворення даних вихідних таблиць у вихідні документи. Як вихідних документів (таблиць) виступають запити і звіти.

При розробці web-додатків необхідно враховувати те, що їх інтерфейс буде показуватися в вікні браузера і, отже, обмежений його можливостями. Універсальним засобом створення уніфікованого інтерфейсу web-сайтів є мова CSS, здатний значно розширити базові можливості мови HTML, пов'язані з позиціонуванням елементів всередині сторінки.

4.2 Інтерфейс авторизації

Аутентифікація - це процес визначення особистості користувача. Цей процес, включає в себе порівняння введеного користувачем ім'я і пароля з паролем, збереженим в базі даних користувачів; У процесі створення елементів сторінки, використовувалися корисні можливості HTML5. Полю, куди користувач буде вводити свою пошту, було присвоєно тип email, це дозволяє браузеру перевірити, ввів чи користувач правильний формат адреса електронної пошти. так само використовувався атрибут required. Елементи, які містять атрибут required є обезательно для заповнення, браузери, які підтримують цей атрибут, не дозволять користувачеві відправити форму поки це поле не буде заповнено, без використання JavaScript.

Відповідно до спроектованим шаблоном був реалізований макет авторизації представлений на рисунках 4.1 та 4.2.

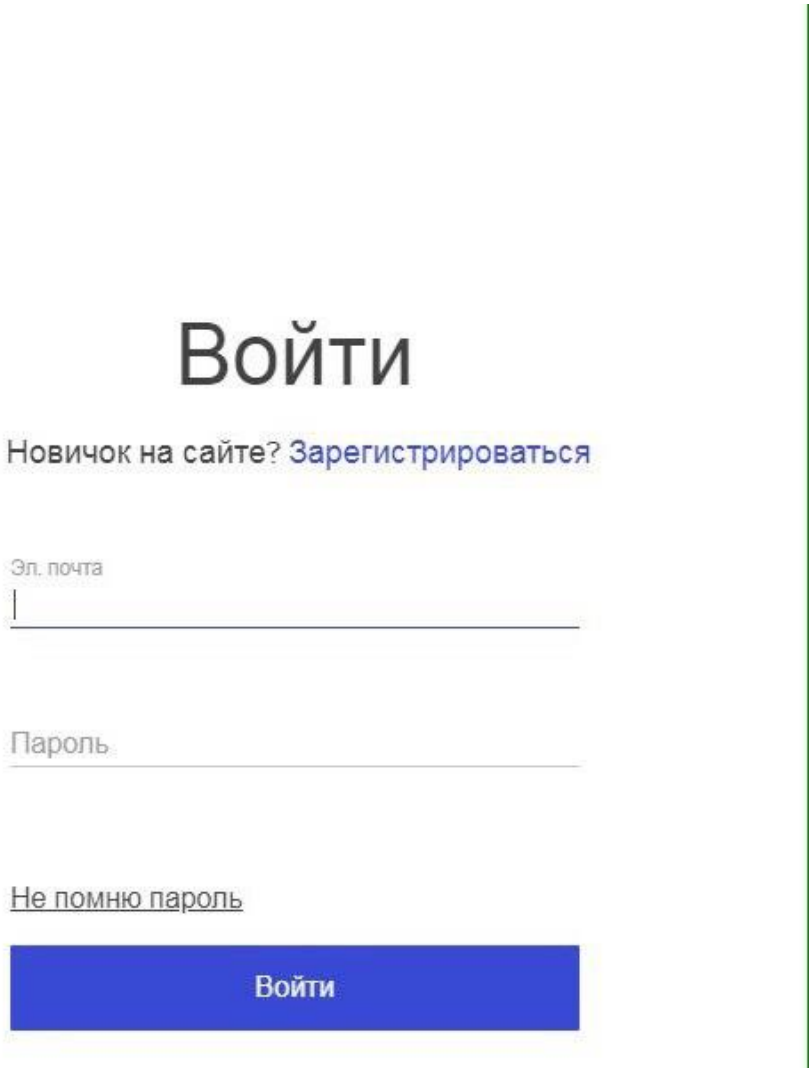

Рисунок 4.1 – Вхід до Веб-застосунку

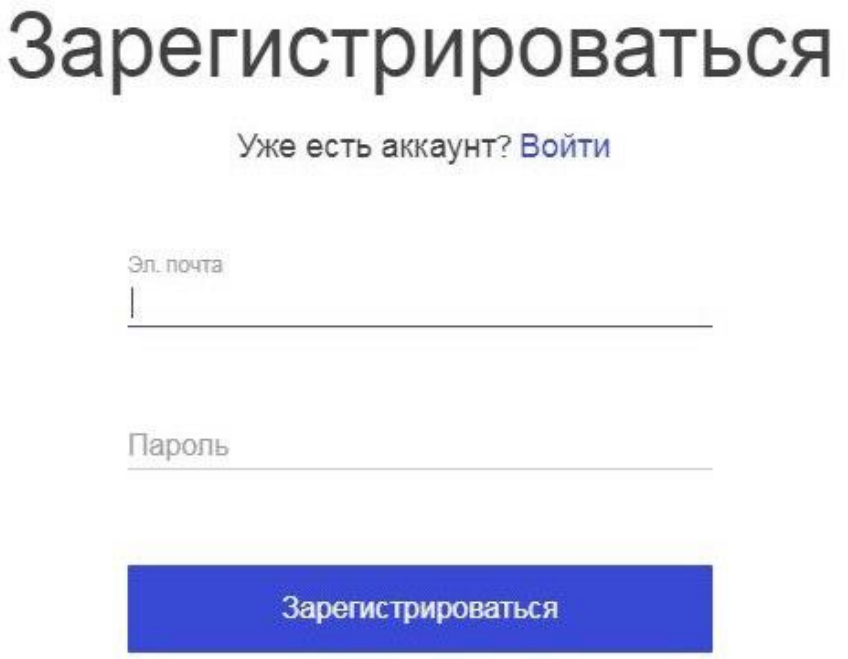

Рисунок 4.2 – Реєстрація у Веб-застосунку

На поштову скриньку адміністратора приходе повідомлення, якщо хтось хоче зареєструватись у Веб-застосунку (Рисунок 4.3). Адміністратор може підтвердити або відмінити запит на реєстрацію. Також, адміністратор бачить список користувачів Веб-застосунком (Рисунок 4.4).

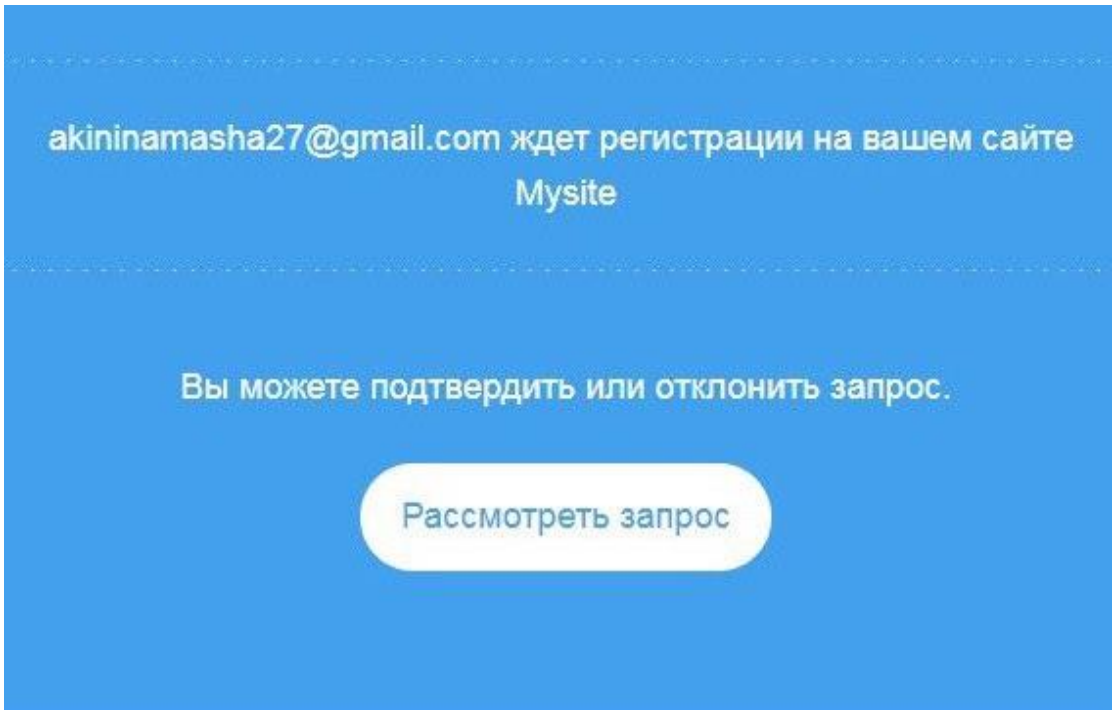

Рисунок 4.3 – Лист-запит реєстрації нового користувача.

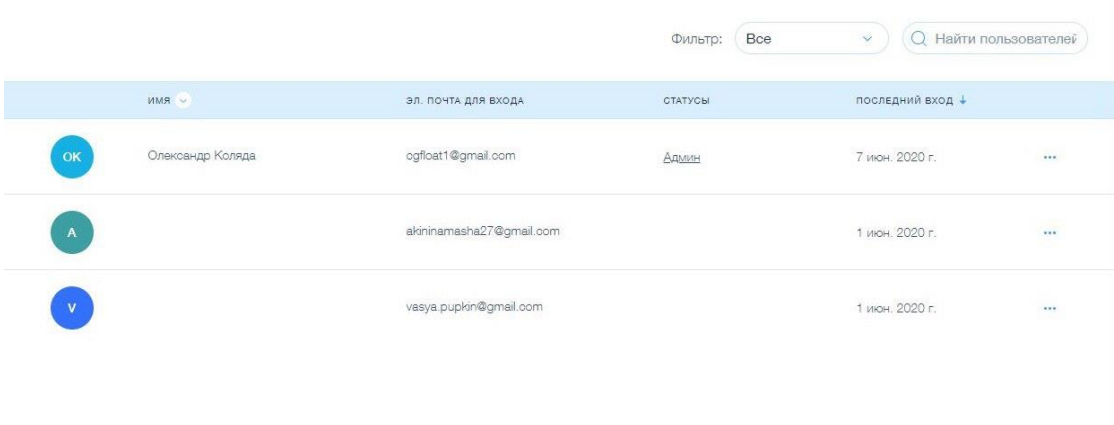

Рисунок 4.4 – Список зареєстрованих користувачів Веб-застосунку.

4.3 Інтерфейс головної сторінки

Сторінки Веб-застосунку мають деякий фіксований зміст, такий як меню, заголовок (шапка), поля для відображення інформації, для цього було вирішено створити основний шаблон сторінки (рисунок 4.5). Основний шаблон (layout) - це незмінна частина сторінки, розташування елементів і блоків щодо один одного в рамках сторінки або інтерфейсу, що включає в себе основні статичні

частини сайту: шапку, меню, підвал. Структура вмісту задається за допомогою HTML, для оформлення зовнішнього вигляду контенту використовується CSS.

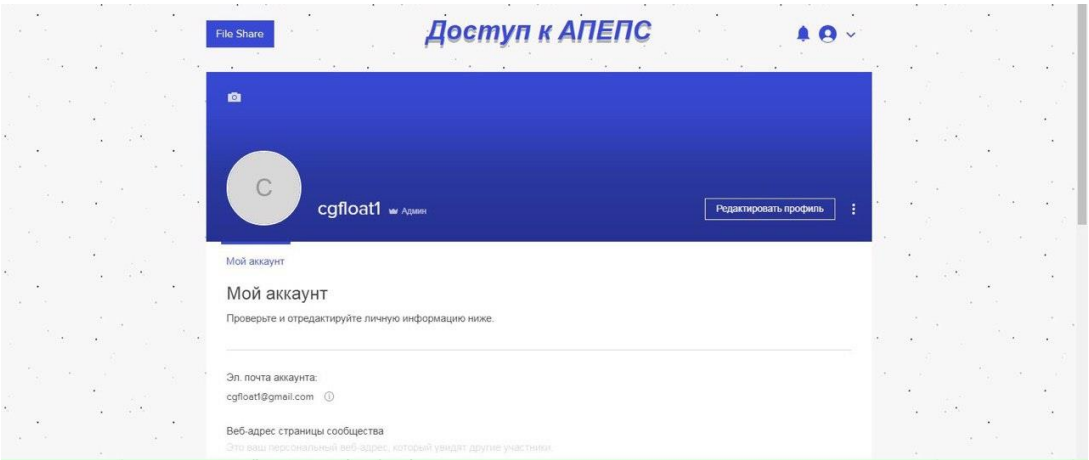

Рисунок 4.5 – Головна сторінка Веб-застосунку

4.4 Сторінка File Share

По кнопці File Share користувач може перейти до свого файлового сховища. У файловому сховищі користувач може додавати, видаляти, зберігати, відкривати та переглядати різні файли (Рисунки 4.6 та 4.7). Типи форматів файлів які можна завантажувати:

- Документи: .doc, .docx, .xls, .xIsx, .ppt, .pptx, .odt, .odp, -pdf
- Зображення: jpg. .png, gif Векторні зображення: .svg
- Шрифти: .tf, .otf, .woff2, .woff
- MP3 Музика: WAV, FLAC, M4A, mp3
- Відео (QuickTime, AVI, MP4, etc.): .avi , .mpeg, .mpg, .mpe, .mp4, .mkv, .webm, .mov, .ogv, .vob, .m4v, .3дp, .divx, .xvid.

Обмеження за розмірами файлів:

- Документи: 25MB
- Зображення: 25MB
- Векторні зображення: 250KB
- Музика: 360MB
- Відео: 15GB на відео або 10 хвилин максимум.

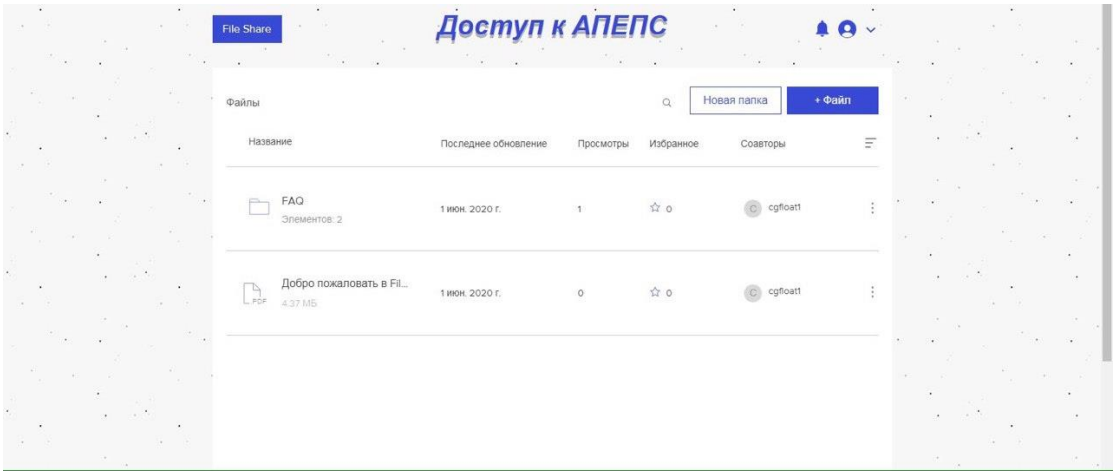

Рисунок 4.6 – Сторінка File Share

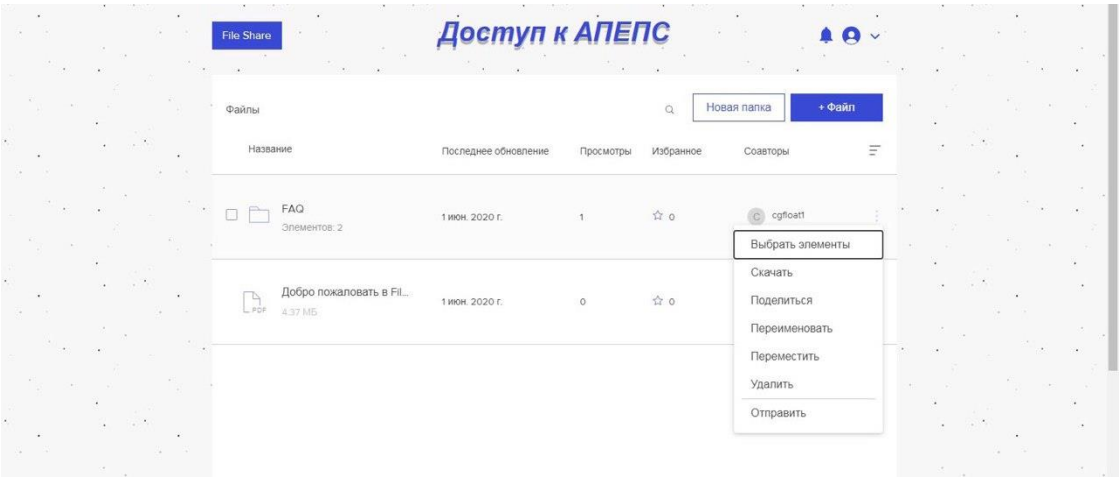

Рисунок 4.7 – Функції з файлами на сторінці File Share

Система, яка створюється, буде розроблятись по принципам клієнт-серверної архітектури, користувач репрезентуватиме одиноку підсистему, що виконуватиме власні функції через взаємодію із користувацьким інтерфейсом, що даватиме сервер Веб-застосунку. Користувацька частина взаємодіятиме із сервером за допомогою веб-інструменту, який розроблений на мові програмування Java. Користувач взаємодіє із Веб-застосунком через http протокол, застосовуючи необхідні запити протоколу. Загальна схема представлена на рисунку 4.8.

Рисунок 4.8 – Принцип взаємодії клієнт-серверної мережі

Цей інтерфейс забезпечує користування Веб-застосунком без установки жодного зайвого програмного забезпечення на портативний комп'ютер користувача. Шлях к системі здійснюється через взаємодію поміж сервером кафедри і портативним комп'ютером користувача. Завдяки цьому, понижаються потреби до технічних характеристик портативного комп'ютера користувача, на якому здійснюватиметься шлях. У майбутньому цей Веб-застосунок можна доповнити іншими користувацькими функціями задля створення багато-функціонального застосунку.

4.5 Висновки з розробки інтерфейсу користувача

У цьому розділі відтворлено базисні моменти розробки інтерфейсу, що позволяє користувачу взаємодіяти з Веб-застосунком. Додано можливість додавати, зберігати, видаляти та переглядати дані, які знаходяться на сервері Веб-застосунку.

Інтерфейс користувача розроблено в вигляді веб-сторінок із застосовуванням HTML, CSS, JavaScript, PHP і також технології для роботи з файловим сховищем даних (File Share). Взаємодіяння інтерфейсу користувача з серверною частиною здійснюється за допомогою використання HTTP-запитів, у центр котрих вводиться потрібна інформація.

Також, інтерфейс користувача дає можливість відновлювати зразки файлів, які перебувають на сервері Веб-застосунку. Тож, Створений інтерфейс користувача позволяє застосовувати потенціал серверу, взаємодіяти з файловим сховищем і розроблений в приємному дизайні з практичною, логічною та ясною структурою застосунку.

ВИСНОВКИ

В дипломній роботі виконувалась розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри. Задля здійснення задачі в першу чергу було виконано розбір базових документів, які впливають на будову академічного процесу кафедри. Також досліджено питання побудови роботи з даними кафедри. Розроблено рекомендації які повинен мати інструмент збереження файлів кафедри.

Подальшим кроком була розробка на базі засобів збереження файлів. Було помічено, що індивідуальні способи збереження даних, як HDD, SSD та flash накопичувачі не підходять потребам працівників. Проблеми вимагають комплексного вирішення, що перероблюють дані кафедри в одну систему, що зобов'язана функціонувати, структурувати і вберігати обсяги даних кафедри.

Було вивчено техніки, що застосовуються задля розробки новітніх систем збереження даних. Досліджено базові принципи предметних моделей, що застосовуються задля будови сховищ даних. Розібрано застосовування наборів віртуалізації задля розподілу програмних технологій правління засобів збереження даних та їх сховища.

Базою побудови системи була реалізація розробки інтерфейсу користувача. Виготовлений інтерфейс користувача підходить до клієнт-серверної архітектури, працює з сервером через http запити. У процесі виготовлення використовувались новітні та потужні технології веб-програмування, завдяки яким створився безперечний та логічний інтерфейс користувача, що допомагає оброблювати використання функцій з даними сервера кафедри.

В даній дипломній роботі створено інструментальні Веб-засоби організації доступу до інформаційних ресурсів кафедри, що мають твердий базис задля майбутнього поширення функціоналу і оптимізації застосунку.

ПЕРЕЛІК ПОСИЛАНЬ

1. Kapadia, A. Implementing cloud storage with Openstack swift: Design, implement, and successfully manage your own cloud storage cluster using the popular OpenStack swift software. / Kapadia, A., Varma, S., Rajana, K. // United Kingdom: Packt Publishing, 2014 – pp. 96 – 129.
2. Toigo, J.W. The holy Grail of enterprise data storage. / Toigo, J.W. // Prentice-Hall, 1999 – pp. 31 – 65.
3. Marc Farley. Rethinking Enterprise Storage: A Hybrid Cloud Model. / Marc Farley. // Microsoft Press, 2013 – pp. 19 – 27.
4. Visual Studio Code [Електроний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Visual_Studio_Code.
5. Database [Електронний ресурс] // Wikipedia. – 2017. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Database.
6. DB-Engines Ranking [Електронний ресурс] // DB-engines. – 2017. – Режим доступу до ресурсу: https://db-engines.com/en/ranking.
7. Hartl, Michael. Ruby on Rails Tutorial: Learn Web Development with Rails. / Hartl, Michael. //Addison-Wesley, 2016 – pp. 137 – 171.
8. Hypertext Transfer Protocol overview [Електронний ресурс] – Режим доступу: https://www.w3.org/Protocols/.
9. Принцип MVC в веб программировании [Електронний ресурс] – Режим доступу: http://folkprog.net/printsip-mvc-u-web-programmirovanii/.
10. Kapadia, Amar, Kris Rajana, Sreedhar Varma. OpenStack Object Storage (Swift) Essentials. / Kapadia, Amar, Kris Rajana, Sreedhar Varma // Packt Publishing, 2015 – pp. 84 – 98.
11. Glover F. Future paths for Integer Programming and Links /Glover F. // Computers and Operations Research. – 1986. – Vol. 5. – P. 378-386.
12. Kirkpatrick S. Optimization by Simulated Annealing / S. Kirkpatrick, C.D. Jr. Gelatt and M.P. Vecchi // Science. – 1983. – Vol. 220, № 4598. – P. 232-248.

ДОДАТОК А

Розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри

Специфікація

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ6288_20Б

Аркушів 1

Київ-2020

Позначення	Найменування	Примітки
Документація		
УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ6288_20Б	Записка.docx	Текстова частина дипломної роботи

ДОДАТОК Б

Розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри

Лістинг програми

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ6288_20Б

Аркушів 7

Київ-2020

Index.html

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
 charset=UTF-8">
</head>
<body>
<div class="line-gutter-backdrop">
</div>
<table>
<tbody>
<tr>
<td class="line-number" value="1">
</td>
<td class="line-content">
<span class="html-doctype">&lt;
 !DOCTYPE html;
</span>
</td>
</tr>
<tr>
<td class="line-number" value="2">
</td>
<td class="line-content"><span class="html-tag">&lt;
html <span class="html-attribute-name">lang
</span>="<span class="html-attribute-value">ru</span>"&gt;
</span>
</td>
</tr>
<tr>
<td class="line-number" value="3">
</td>
<td class="line-content"><span class="html-tag">&lt;
head&gt;
</span>
</td>
</tr><tr>
<td class="line-number" value="4"></td><td class="line-content">
<span class="html-tag">&lt;
meta <span class="html-attribute-name">http-equiv</span>="<span class="html-
attribute-value">X-UA-Compatible</span>
 <span class="html-attribute-name">content</span>="<span class="html-attribute-
value">IE=Edge</span>"/&gt;
```

```
</span>
</td>
</tr>
<tr>
<td class="line-number" value="5"></td><td class="line-content">
 </td>
</tr>
<tr>
<td class="line-number" value="6">
</td><td class="line-content">
<span class="html-tag">&lt;
meta <span class="html-attribute-name">charset</span>
<span class="html-attribute-value">utf-8</span>"/&gt;</span></td></tr><tr><td
class="line-number" value="7">
</td><td class="line-content">    <span class="html-tag">&lt;
meta <span class="html-attribute-name">name</span>="<span class="html-attribute-
value">generator</span>" <span class="html-attribute-name">content</span>="<span
class="html-attribute-value">Wix.com Website Builder</span>"/&gt;
</span>
</td>
</tr>
<tr>
<td class="line-number" value="8">
</td>
<td class="line-content">
 <span class="html-tag">&lt;
link <span class="html-attribute-name">rel</span>="<span class="html-attribute-
value">shortcut icon</span>" <span class="html-attribute-name">href</span>="<a
class="html-attribute-value html-resource-link" target="_blank"
href="https://www.com/favicon.ico" rel="noreferrer
noopener">https://www.com/favicon.ico</a>"
<span class="html-attribute-name">type</span>="<span class="html-attribute-
value">image/x-icon</span>"/&gt;
</span></td></tr><tr><td class="line-number" value="9"></td><td class="line-
content">    <span class="html-tag">&lt;
link <span class="html-attribute-name">rel</span>="<span class="html-attribute-
value">apple-touch-icon</span>"
<span class="html-attribute-name">href</span>="<a class="html-attribute-value html-
resource-link" target="_blank" href="https://www.com/favicon.ico" rel="noreferrer
noopener">https://www.com/favicon.ico</a>" <span class="html-attribute-
name">type</span>="<span class="html-attribute-value">image/x-icon</span>"/&gt;
</span>
</td></tr>
<tr>
```

<td class="line-number" value="10"></td>
<td class="line-content">
</td></tr>
<tr>
<td class="line-number" value="11">
</td><td class="line-content">
</td></tr>
<tr>
<td class="line-number" value="12">
</td><td class="line-content">
</td></tr>
<tr>
<td class="line-number" value="13">
</td><td class="line-content">
</td></tr>
<tr>
<td class="line-number" value="14"></td>
<td class="line-content"> <
meta http-equiv="X-Wix-Meta-Site-Id" content="8bc282ca-ee94-4828-a677-2b5cadc47368"/>

</td></tr>
<tr>
<td class="line-number" value="15"></td><td class="line-content"> <
meta http-equiv="X -Application-Instance-Id" content="d52d9192-ce8f-410b-83ea-1a894d83d485"/>

</td></tr>
<tr>
<td class="line-number" value="16">
</td>
<td class="line-content">
</td></tr>
<tr><td class="line-number" value="17">
</td><td class="line-content"> </td></tr>
<tr><td class="line-number" value="18"></td>
<td class="line-content">
  <

meta http-equiv="X-Wix-Published-Version" content="32"/>
</td></tr><tr><td class="line-number" value="19">
</td><td class="line-content">   </td></tr><tr><td class="line-number" value="20">
</td><td class="line-content">   </td></tr><tr><td class="line-number" value="21">
</td><td class="line-content">   </td></tr><tr><td class="line-number" value="22">
</td><td class="line-content">   </td></tr><tr><td class="line-number" value="23">
</td><td class="line-content">   <
        meta name = "format-detection" content = "telephone=no"/>
    </td></tr><tr><td class="line-number" value="24"></td><td class="line-content">   </td></tr><tr><td class="line-number" value="25"></td><td class="line-content">   </td></tr><tr><td class="line-number" value="26"></td><td class="line-content">
</td></tr><tr><td class="line-number" value="27"></td><td class="line-content">   <
        meta name="SKYPE_TOOLBAR" content="
_TOOLBAR_PARSER_COMPATIBLE"/>

</td></tr>
<tr><td class="line-number" value="28"></td><td class="line-content">
</td>
</tr><tr>
<td class="line-number" value="29"></td><td class="line-content">   </td></tr><tr>
<td class="line-number" value="30"></td><td class="line-content">   <meta id="wixMobileViewport" name="viewport" content="width=980, user-scalable=yes"/>
</td></tr><tr><td class="line-number" value="31"></td><td class="line-content">
    </td></tr>
<tr><td class="line-number" value="32"></td><td class="line-content">
</td>
</tr><tr>
        <td class="line-number" value="33"></td><td class="line-content">
</td></tr>
<tr><td class="line-number" value="34"></td><td class="line-content">
        </td></tr>
<tr><td class="line-number" value="35"></td><td class="line-content">
        </td></tr>

```
<tr><td class="line-number" value="36">
</td><td class="line-content">
                </td></tr>
<tr><td class="line-number" value="37">
</td><td class="line-content">
                    <br></td>
</tr><tr><td class="line-number" value="38">
</td><td class="line-content">
                    </td></tr><tr><td class="line-number" value="39">
</td><td class="line-content">
                        <br></td>
</tr><tr><td class="line-number" value="40">
</td><td class="line-content">
                            <br></td>
</tr><tr><td class="line-number" value="41">
</td><td class="line-content">
                                <br></td>
</tr><tr><td class="line-number" value="42">
</td><td class="line-content">
                                    <br></td>
</tr><tr><td class="line-number" value="43">
</td><td class="line-content">
                                        <br></td>
</tr><tr><td class="line-number" value="44">
</td><td class="line-content">
<span class="html-comment">&lt;
        !-- META DATA --&gt;
</span></td></tr><tr><td class="line-number" value="45"></td><td class="line-
content">    <span class="html-tag">&lt;
        script <span class="html-attribute-name">type</span>="<span class="html-
attribute-value">text/javascript</span>"&gt;
        </span></td></tr><tr><td class="line-number" value="46"></td><td
class="line-content">        </td></tr>
<tr><td class="line-number" value="47"></td><td class="line-content"><br></td>
</tr><tr><td class="line-number" value="48"></td><td class="line-content">

        </td></tr><tr><td class="line-number" value="49"></td><td class="line-
content">        var santaModels = true;
</td></tr><tr><td class="line-number" value="50">
</td><td class="line-content">        var isStreaming = true;


</td></tr><tr><td class="line-number" value="52"></td><td class="line-content">
</td></tr><tr><td class="line-number" value="53">
```

```
</td><td class="line-content"><br></td></tr><tr><td class="line-number"
value="54"></td><td class="line-content">       var googleAnalytics = ""
</td></tr><tr><td class="line-number" value="55">
</td><td class="line-content">       var ipAnonymization = false
</td></tr><tr><td class="line-number" value="56">
</td><td class="line-content"><br></td></tr><tr><td class="line-number"
value="57"></td><td class="line-content">       var googleRemarketing = ""
</td></tr><tr><td class="line-number" value="58">
</td><td class="line-content">       var googleTagManager = ""
</td></tr><tr><td class="line-number" value="59">
</td><td class="line-content">       var facebookRemarketing = ""
</td></tr><tr><td class="line-number" value="60">
</td><td class="line-content">       var yandexMetrika = ""
</td></tr><tr><td class="line-number" value="61">
</td><td class="line-content">    <span class="html-tag">&lt
/script&gt
</span></td></tr><tr><td class="line-number" value="62">
</td><td class="line-content">    </td></tr><tr><td class="line-number"
value="63"></td><td class="line-content">    </td></tr><tr><td class="line-number"
value="64"></td><td class="line-content"><br></td></tr><tr><td class="line-number"
value="65"></td><td class="line-content">    </td></tr><tr><td class="line-number"
value="66"></td><td class="line-content"><br></td></tr><tr><td class="line-number"
value="67"></td><td class="line-content">    <span class="html-tag">&lt
script&gt
</span></td></tr><tr><td class="line-number" value="68">
</td><td class="line-content">       var wixBiSession = {</td></tr><tr><td class="line-
number" value="69"></td><td class="line-content">           initialTimestamp:
Date.now()</td></tr><tr><td class="line-number" value="70">
</td><td class="line-content">           , ssrRequestTimestamp:
1591036670090</td></tr><tr>
<td class="line-number" value="71">
</td><td class="line-content">           , requestId: publicModel.requestId</td>
</tr>
<tr>
<td class="line-number" value="72">
</td>
<td class="line-content">           , genGUID: function () {</td></tr><tr><td
class="line-number" value="73">
</td><td class="line-content">
  return 'xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx'.replace(/[xy]/g,function(c){var
r=Math.random()*16|0,v=c=='x'?r:(r&amp;0x3|0x8)
return v.toString(16)
})
</td></tr>
```

<tr><td class="line-number" value="74">
</td><td class="line-content"> }</td></tr>
<tr><td class="line-number" value="75">
</td><td class="line-content"> , sessionId: 'c068882e-98ac-41ff-a454-
99c80a1cca75'</td></tr><tr><td class="line-number" value="76">
</td><td class="line-content"> , initialRequestTimestamp:
performance.timeOrigin ? performance.timeOrigin : Date.now() -
performance.now()</td></tr><tr><td class="line-number" value="77">
</td><td class="line-content"> </td></tr>
<tr><td class="line-number" value="78">
</td><td class="line-content"> </td></tr>
<tr><td class="line-number" value="79">
</td><td class="line-content"> , siteMemberId: 'ab7624a4-b027-439b-999c-
4a6d429a673a'</td></tr><tr><td class="line-number" value="80">
</td><td class="line-content"> </td></tr><tr><td class="line-number"
value="81"></td><td class="line-content"> , is_rollout: 0</td></tr><tr><td
class="line-number" value="82">
</td><td class="line-content"> , isSAVrollout: 0</td></tr><tr><td class="line-
number" value="83"></td><td class="line-content"> , is_platform_loaded:
1</td></tr><tr><td class="line-number" value="84">
</td><td class="line-content"> , suppressbi: false</td></tr><tr><td class="line-
number" value="85"></td><td class="line-content"> , dc: '84'</td></tr><tr><td
class="line-number" value="86">
</td><td class="line-content"> , renderType: 'bolt'</td></tr><tr><td class="line-
number" value="87">
</td><td class="line-content"> , siteRevision: '32'</td></tr><tr><td class="line-
number" value="88"></td><td class="line-content"> , siteCacheRevision:
'1591036661586'</td></tr><tr><td class="line-number" value="89">
</td><td class="line-content"> , wixBoltExclusionReason: "</td></tr><tr><td
class="line-number" value="90">
</td><td class="line-content"> , wixBoltExclusionReasonMoreInfo:
"</td></tr><tr><td class="line-number" value="91">
</td><td class="line-content"> , checkVisibility: (function () {</td></tr><tr><td
class="line-number" value="92"></td><td class="line-content"> var
alwaysVisible = document.hidden !== true
</td></tr><tr><td class="line-number" value="93">
</td><td class="line-content"> function checkVisibility() {</td></tr><tr><td
class="line-number" value="94"></td><td class="line-content">
alwaysVisible = alwaysVisible &amp

ДОДОТОК В

Розробка інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри

Опис програми

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ6288_20Б

Аркушів 8

Київ-2020

АНОТАЦІЯ

Цей додаток включає в собі опис основних компонентів інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри, а саме Веб-сайт для робітників кафедри.

Застосунок зроблений за допомогою таких технологій, як HTML, CSS, JavaScript, File Share.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ

У цьому додатку міститься опис основних компонентів інструментальних Веб-засобів організації доступу до інформаційних ресурсів кафедри.

В додатку Б знаходиться програмний код застосунку.

Застосунок зроблений за допомогою таких технологій, як HTML, CSS, JavaScript, File Share.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Додаток надає функціональність для роботи з інструментальними Веб-засобами організації доступу до інформаційних ресурсів кафедри.

— створення, зберігання, редагування та видалення файлів на файловому сховищі;

— реєстрація користувачів на сайті;

— сортування файлів по папкам;

— авторизація користувачів на сайті.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Згідно з архітектурою, додаток складається з серверної частини, власної сторінки для користувача сайту та сторінки для адміністратора сайту.

На сайті є такі розділи як інформаційна сторінка користувача, де вказані його персональні дані, такі як прізвище, ім’я, E-mail та номер телефону. Також є розділ файлового сховища даних, де користувач може створювати, зберігати, редагувати та видаляти файли.

Адмінстратор сайту має змогу дозволяти або блокувати запити на реєстрацію нових користувачів сайту, бачити список всіх поточних користувачів сайту.

ВИКОРИСТАНІ ТЕХНІЧНІ ЗАСОБИ

Для використання сайту користувач повинен мати обліковий запис на електронній поштовій скринці, портативний компь’ютер, планшет або мобільний телефон, а також підключення до мережі Інтернет.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для того щоб користуватись функціями сайту необхідно перейти на електронну адресу сайту на своєму пристрої, якій підключений до мережі Інтернет та зареєструватись. Адміністратор сайту повинен дозволити запит на реєстрацію користувача. Після цього користувач може авторизуватись на сайті та користуватись функціями.

ВХІДНІ ТА ВИХІДНІ ДАНІ

Вхідними даними для цього сайту є авторизація, а також додавання файлів до файлового сховища. Вихідними данними є список файлів які були додані раніше та дані про користувача облікового запису, зареєстрованого та авторизованого на сайті.